



UNIVERSIDADE DA BEIRA INTERIOR
Engenharia

Controlo e Automação Sistema de rega inteligente

Tiago André Coelho Soares

Dissertação para obtenção do Grau de Mestre em
Engenharia Electrotécnica e de Computadores:
Automação e Electrónica
(2º ciclo de estudos)

Orientador: Prof. Doutor Sérgio José Pinto Simões Mariano

Covilhã, Junho de 2012

Agradecimentos

- À minha família e namorada, por todo o apoio e força que me deram;
- À empresa TecnoSoares, Lda, pela oportunidade de estágio, conhecimento e experiência adquirida ao longo da realização do mesmo;
- Ao Professor Doutor Sílvio José Pinto Simões Mariano, pelo seu apoio, vasto conhecimento e ajuda ao longo de toda a realização do projecto;
- À UBI, pelo material disponibilizado e pela oportunidade e hospitalidade com que me recebeu, permitindo que terminasse mais uma fase da minha vida e fornecendo o devido conhecimento e ferramentas para conseguir evoluir futuramente, tanto a nível profissional como pessoal;
- A todos os meus colegas da UBI e intervenientes que me apoiaram e ajudaram ao longo deste trabalho.

Muito obrigado a todos!

Prefácio

A presente dissertação de mestrado surgiu no âmbito da disciplina de Dissertação, com o intuito de descrever o trabalho efectuado ao longo do passado ano lectivo.

A escolha do tema derivou de um gosto pessoal relativo ao controlo e automação de sistemas recorrendo a Controladores Lógicos Programáveis (*Programmable Logic Controllers* - PLC's) e consolas HMI.

O assunto desta dissertação assenta na implementação de um sistema de rega inteligente recorrendo a um PLC.

O objectivo é criar um sistema totalmente autónomo e com capacidade para preencher todas as lacunas presentes num sistema manual do mesmo tipo.

Este sistema será posteriormente aplicado a um projecto real.

Resumo

Nesta dissertação consta a apresentação de um sistema que permite o controlo e monitorização de um sistema de abastecimento de água para fins de rega.

Inicialmente, começa por ser feita uma introdução acerca dos PLC's e a sua utilização nas diversas áreas. Segue-se uma descrição dos objectivos, assim como os factores que levaram à realização deste projecto. Termina-se o capítulo com a apresentação da estrutura utilizada nesta dissertação.

No capítulo seguinte é apresentado o conceito geral dos PLC's. Ainda dentro do mesmo capítulo, segue-se a evolução histórica ao longo dos anos, a descrição, a explicação dos diversos tipos de PLC's e suas partes constituintes. O capítulo termina com a apresentação e análise das diferentes linguagens de programação existentes.

No terceiro capítulo é apresentado o conceito geral do sistema. Posteriormente, é apresentado o *hardware* e *software* utilizados para a conclusão do projecto. Finaliza-se o capítulo com a apresentação detalhada do sistema e eventuais melhorias futuras do mesmo.

A dissertação termina com comentários e conclusões relativamente à realização deste projecto e trabalho, assim como a devida bibliografia e anexos referentes ao mesmo.

Palavras-chave

Controlo, monitorização, sistema de abastecimento de água e PLC.

Abstract

This dissertation contains the presentation of a system that allows the control and monitoring of a water supply system to be used for irrigation.

Initially, it begins with an introduction of the PLC's and their use in several areas. It follows a description of goals, as well as the factors that led to this project. The chapter ends with the presentation of the structure used in the dissertation.

The next chapter presents the general concept of PLC's. Still in the same chapter, it follows the historical evolution over the years, the description and explanation of the various types of PLC's and its parts. The chapter concludes with a presentation and analysis of the various programming languages available.

The third chapter presents the general concept of the system. Subsequently, the used hardware and software is presented. The chapter ends with a detailed presentation of the system and possible future improvements of the work.

The dissertation concludes with comments and conclusions regarding the implementation of this project and work, as well as the proper bibliography and attachments referring to it.

Keywords

Control, monitoring, water supply system and PLC.

Índice

1.Capítulo 1	1
1.1. Introdução	1
1.1.1. Contextualização	5
1.1.2. Objectivos	5
1.1.3. Organização da dissertação	6
2.Capítulo 2	7
2.1. PLC's	7
2.1.1. O que é e como funciona um PLC?	7
2.1.2. História da automação	9
2.1.3. Evolução	10
2.1.4. Partes constituintes de um PLC	12
2.1.4.1. Entradas digitais/discretas	12
2.1.4.2. Entradas analógicas	18
2.1.4.3. Saídas digitais/discretas	21
2.1.4.4. Saídas analógicas	26
2.1.4.5. Processadores (CPU's)	27
2.1.4.6. Memória	29
2.1.4.7. Alimentação	38
2.1.4.8. Dispositivos de programação	40
2.1.4.9. Interfaces de operação	43
2.1.4.10. Linguagens de programação	44
2.1.4.10.1. NORMA IEC 1131-3	45
2.1.4.10.2. LINGUAGEM LADDER	46
2.1.4.10.3. LINGUAGEM BOOLEANA	53
2.1.4.10.4. LINGUAGEM GRAFCET	56
2.1.4.10.5. DIAGRAMAS DE BLOCOS DE FUNÇÕES	58
2.1.4.10.6. LISTA DE INSTRUÇÕES	59
2.1.4.10.7. TEXTO ESTRUTURADO	60
3.Capítulo 3	62
3.1. Conceito do sistema	62
3.1.1. Características gerais	63
3.1.2. Introdução ao Siemens S7-1200 1212C	65
3.1.2.1. Módulos e especificações	67
3.1.3. Introdução ao <i>software</i> de programação	70
3.1.4. Introdução à consola KTP600 Basic Mono PN	72

3.1.5.	Detalhes do sistema	73
3.1.5.1.	Programa de controlo.....	75
3.1.5.2.	Implementação da base de dados	80
3.1.5.3.	Programa da consola HMI.....	84
3.1.5.4.	Criação da página <i>web</i>	86
3.1.6.	Potencialidades do sistema.....	88
3.1.6.1.	Comunicação GSM/GPRS	88
3.1.6.2.	Monitorização do solo via <i>ZigBee</i> e GPRS.....	90
3.1.6.3.	Sincronização do relógio interno via GPS.....	92
3.1.6.4.	Sincronização do relógio interno via servidor NTP.....	93
4.	Conclusão	94
4.1.	Trabalhos futuros	95
5.	Referências.....	96
6.	Anexos.....	99
6.1.	Anexo I	99
6.2.	Anexo II	105

Lista de Figuras

Figura 1.1 - Exemplo de um processo onde é utilizado o PLC	1
Figura 1.2 - Outro exemplo da utilização do PLC	1
Figura 1.3 - Trabalho manual.....	2
Figura 1.4 - Trabalho automático.....	2
Figura 1.5 - Exemplo de uma linha de montagem onde a interacção humana é mínima	2
Figura 1.6 - Fluxo de um sistema de controlo	5
Figura 2.1 - Diagrama da aplicação do PLC a um determinado processo ou máquina.....	7
Figura 2.2 - Constituição de um PLC	8
Figura 2.3 - Componentes constituintes da CPU	8
Figura 2.4 - Diagrama de entradas AC/DC.....	13
Figura 2.5 - Circuito eléctrico de entradas AC/DC.....	14
Figura 2.6 - Exemplo de uma entrada <i>sinking</i> (a) Exemplo de uma entrada <i>sourcing</i> (b)	15
Figura 2.7 - Ligação de um módulo de entradas isoladas AC/DC.....	16
Figura 2.8 - Ligação de um módulo de entradas TTL.....	16
Figura 2.9 - <i>Interface</i> BCD a armazenar dados numa <i>word</i> /registo de memória.....	17
Figura 2.10 - Diagrama exemplificativo da multiplexagem num módulo de entrada BCD	17
Figura 2.11 - Exemplo de um sinal analógico medido	18
Figura 2.12 - Sensores analógicos de humidade (esquerda) e temperatura (direita).....	18
Figura 2.13 - Característica ideal de um conversor A/D com resolução de 3 <i>bits</i>	19
Figura 2.14 - Exemplos de <i>interfaces</i> de expansão de entradas	20
Figura 2.15 - Exemplo de um circuito de isolamento óptico de um PLC.....	21
Figura 2.16 - Esquema simples da constituição de um relé	21
Figura 2.17 - Representação de transístores PNP e NPN e respectivas simbologias	22
Figura 2.18 - Representação de um triac (esquerda) e a sua simbologia (direita).....	22
Figura 2.19 - Diagrama de entradas AC	23
Figura 2.20 - Circuito típico de saídas AC	24
Figura 2.21 - Circuito típico de saídas DC	24
Figura 2.22 - Diagrama de ligação para uma <i>interface</i> de saídas AC isoladas	25
Figura 2.23 - Informação de 16 <i>bits</i> a ser enviada para um módulo de saída.....	25
Figura 2.24 - Representação de um circuito capaz de ajustar uma válvula de volume	26
Figura 2.25 - Conversão de um valor digital num valor analógico	26
Figura 2.26 - Representação do processo de <i>scan</i> de um PLC	28
Figura 2.27 - Exemplo de um sinal de entrada que não será detectado pelo PLC	28
Figura 2.28 - Exemplo de um <i>chip</i> de memória para um PLC	30

Figura 2.29 - Exemplo de memórias ROM.....	30
Figura 2.30 - Exemplo de memória RAM.....	31
Figura 2.31 - Exemplo de memória PROM	31
Figura 2.32 - Exemplo de memória EPROM.....	32
Figura 2.33 - Estrutura de uma <i>word</i> de memória	33
Figura 2.34 - Ilustração de um bloco de 4K com armazenamento de 8 <i>bits</i> (a) Ilustração de um bloco de 4K com um armazenamento de 16 <i>bits</i> (b).....	34
Figura 2.35 - Mapa de memória simplificado	34
Figura 2.36 - Mapa da memória de aplicações.....	35
Figura 2.37 - Alteração de um <i>bit</i> na tabela de entradas	36
Figura 2.38 - Alteração de um <i>bit</i> na tabela de saídas.....	37
Figura 2.39 - Área de armazenamento da tabela de dados	38
Figura 2.40 - Ligação de um transformador de tensão constante e um PLC	40
Figura 2.41 - PC como ponte de interligação entre uma <i>mainframe</i> e uma rede de PLC's	42
Figura 2.42 - Exemplo de um terminal portátil de programação.....	43
Figura 2.43 - Circuito <i>ladder</i> físico vs circuito <i>ladder</i> no PLC	46
Figura 2.44 - Simbologia básica da linguagem <i>ladder</i>	47
Figura 2.45 - Estrutura de uma <i>rung</i>	51
Figura 2.46 - Caminhos de continuidade numa <i>rung</i>	51
Figura 2.47 - Lista das funções mais utilizadas no <i>software</i> STEP 7 V11.....	52
Figura 2.48 - Programa em linguagem booleana e correspondente em <i>ladder</i>	53
Figura 2.49 - Relação da álgebra booleana com as funções <i>AND</i> , <i>OR</i> e <i>NOT</i>	54
Figura 2.50 - Portas básicas da lógica booleana.....	54
Figura 2.51 - Combinação das portas lógicas básicas booleanas	54
Figura 2.52 - Operações lógicas utilizando álgebra de Boole	55
Figura 2.53 - Programa em <i>grafcet</i> e conversão para <i>ladder</i>	57
Figura 2.54 - Comparação entre linguagem <i>grafcet</i> (a) e linguagem SFC (b)	58
Figura 2.55 - Exemplo de programa em FBD	58
Figura 2.56 - Bloco de função programado em <i>ladder</i> (a) Bloco <i>Start/Stop</i> em FBD (b).....	59
Figura 2.57 - Bloco de função em linguagem de lista de instruções.....	60
Figura 2.58 - Exemplo de linguagem BASIC	60
Figura 2.59 - Bloco de função criado utilizando linguagem em texto estruturado.....	61
Figura 3.1 - Esquema do sistema desenvolvido.....	64
Figura 3.2 - Constituição básica do S7-1200	65
Figura 3.3 - Interface <i>Web Server</i>	66
Figura 3.4 - Visualização e <i>download</i> dos <i>data logs</i>	66

Figura 3.5 - Visualização, <i>download</i> e eliminação dos <i>data logs</i> com <i>login</i> de administrador.....	67
Figura 3.6 - Tipos de módulos de expansão permitidos.....	69
Figura 3.7 - Janela inicial do STEP 7 V11	71
Figura 3.8 - Janela de projecto do STEP 7 V11	71
Figura 3.9 - Facilidade na criação de programas.....	71
Figura 3.10 - Constituição da consola HMI.....	72
Figura 3.11 - Módulo de uma saída analógica.....	75
Figura 3.12 - Atribuição do endereço de IP ao PLC	75
Figura 3.13 - Activação de <i>bits</i> do sistema.....	76
Figura 3.14 - Activação do sistema <i>Web Server</i>	76
Figura 3.15 - Função <i>retain</i> dos blocos de dados.	80
Figura 3.16 - Criação e abertura de um <i>data log</i>	80
Figura 3.17 - Criação do registo diário.....	81
Figura 3.18 - Fecho do <i>data log</i> mensal	81
Figura 3.19 - Abertura do <i>data log</i> cada vez que o PLC é reiniciado	82
Figura 3.20 - Atribuição do nome aos <i>data logs</i>	82
Figura 3.21 - Determinação de ano bissexto.....	82
Figura 3.22 - Exemplo devidamente organizado de um <i>data log</i> do mês de Janeiro	83
Figura 3.23 - Adição da consola HMI ao projecto	84
Figura 3.24 - Configuração do IP e <i>network</i> da consola.....	85
Figura 3.25 - Configuração da ligação entre PLC e consola HMI.....	85
Figura 3.26 - <i>Interface</i> de edição do programa <i>Notepad++</i>	86
Figura 3.27 - Páginas <i>web</i> criadas	87
Figura 3.28 - Sistema proposto	90
Figura 3.29 - Sistema proposto detalhado.....	91
Figura 3.30 - Sistema proposto para sincronização do relógio interno do PLC.....	93
Figura 3.31 - Activação da sincronização do relógio interno via servidores NTP	93

Lista de Tabelas

Tabela 2.1 - Valores típicos admissíveis nas <i>interfaces</i> de entradas analógicas	19
Tabela 2.2 - Dispositivos de saída	23
Tabela 2.3 - Tipos de saída	23
Tabela 2.4 - Tipos de saídas analógicas.....	27
Tabela 2.5 - Tipos de variáveis de dados: <i>Bit</i> , <i>Byte</i> , <i>Word</i> e <i>Dword</i>	48
Tabela 2.6 - Tipo de variável de dados: <i>Integer</i>	49
Tabela 2.7 - Tipo de variável de dados: <i>Floating-point real</i>	49
Tabela 2.8 - Tipos de variáveis de dados: <i>Time</i> , <i>Data</i> , <i>Time_of_Day</i> e DTL.....	49
Tabela 2.9 - Tipos de variáveis de dados: <i>Char</i> e <i>String</i>	49
Tabela 2.10 - Instruções booleanas e respectivas instruções em <i>ladder</i>	55
Tabela 3.1 - Características da CPU 1212C.....	68
Tabela 3.2 - Módulos de entradas e saídas digitais.....	69
Tabela 3.3 - Módulos de entradas e saídas analógicas	69
Tabela 3.4 - Módulos de comunicação	70
Tabela 3.5 - Funcionalidades de algumas redes sem fios existentes	89
Tabela 3.6 - Tabela de especificações do bloco de função “gps_rcv”	92

Lista de Acrónimos

AC	Alternating Current
A/D	Analog-to-Digital
BASIC	Beginner's All-purpose Symbolic Instruction Code
BCD	Binary Coded Decimal
CPU	Central Processing Unit
D/A	Digital-to-Analog
DC	Direct Current
DTU	Data Terminal Unit
EAROM	Electrically Alterable Read-Only Memory
EEPROM	Electrically Erasable Programmable Read-Only Memory
EMI	Electromagnetic Interference
EPROM	Erasable Programmable Read-Only Memory
FBD	Function Block Diagram
FDMA	Frequency-Division Multiple Access
FET	Field Effect Transistor
FFD	Full Function Device
FIFO	First In First Out
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile Communication
GUI	Graphical User Interface
HHP	Hand-Held Programmer
HMI	Human Machine Interface
HTML	HyperText Markup Language
IL	Instruction List
I/O	Input/Output
IP	Internet Protocol
ISDN	Integrated Services Digital Network

LD	Ladder Diagram
LED	Light-Emitting Diode
LIFO	Last In First Out
LSB	Least Significant Bit
MAC	Medium Access Control
MOV	Metal Oxide Varistor
MSB	Most Significant Bit
NTP	Network Time Protocol
OS	Operating System
PC	Personal Computer
PLC	Programmable Logic Controller
PROFIBUS	Process Field Bus
PROM	Programmable Read-Only Memory
PWM	Pulse-Width Modulation
RAM	Random-Access Memory
RC	Resistance Condenser
RF	Radio Frequency
RFD	Reduced Function Device
ROM	Read-Only Memory
R/W	Read/Write
SCADA	Supervisory Control And Data Acquisition
SCR	Silicon Controlled Rectifier
SFC	Sequential Function Chart
SMS	Short Message Service
SPM	Scratchpad Memory
ST	Structured Text
TDMA	Time-Division Multiple Access
TTL	Transistor-Transistor Logic

1. Capítulo 1

1.1. Introdução

Os PLC's, ou *Programmable Logic Controllers* (Controladores Lógicos Programáveis), são uma grande aposta para quando se pretende automatizar um processo. Todos os aspectos da indústria, desde a geração de energia, passando pela pintura de automóveis (Fig. 1.1) até ao embalamento de comida (Fig. 1.2), recorrem a PLC's para aumentar e melhorar a qualidade geral de produtos produzidos em massa.



Figura 1.1 - Exemplo de um processo onde é utilizado o PLC [2].



Figura 1.2 - Outro exemplo da utilização do PLC [3].

O interesse neste tipo de dispositivos surgiu da necessidade de tornar um processo o mais eficiente, rentável e autónomo possível.

O conceito de automação assume a ideia da utilização de potência eléctrica ou mecânica para accionar algum tipo de máquina [4]. Como se trata de um processo autónomo, algum tipo de inteligência deverá ser adicionado à máquina.

A palavra automação possui vários significados; contudo, todos eles convergem para o mesmo termo: autónomo. Seguem-se alguns significados da palavra automação [4]:

- Substituição do trabalho humano ou animal por uma máquina (tornar autónomo) (Fig. 1.3);
- Controlo autónomo de um sistema com a mínima interferência do operador humano (Fig. 1.4);
- Mecanismo de actuação própria, que realiza uma determinada acção num determinado tempo ou em resposta a determinadas condições - novamente, retém-se a ideia de um processo automático (Fig. 1.5).



Figura 1.3 - Trabalho manual [5].



Figura 1.4 - Trabalho automático [6].



Figura 1.5 - Exemplo de uma linha de montagem onde a interação humana é mínima [7].

A automação é também considerada como um meio através da qual é possível atingir melhores níveis de qualidade em produtos produzidos em grande quantidade [8]. Hoje em dia, a qualidade não consiste apenas no controlo final do produto; é imprescindível que esteja presente desde o início do fabrico de um determinado produto, quer através de um rigoroso controlo dimensional das grandezas envolvidas quer através de inspecções periódicas para determinar se os padrões ou parâmetros estão de acordo com os pressupostos.

Além de tornar os sistemas mais eficientes, rentáveis, autónomos e com uma qualidade geral superior, a automação visa também aumentar a segurança dos utilizadores ou daqueles que interagem directamente com a máquina.

Assim, as vantagens de usar PLC num determinado processo são as seguintes:

- Diminuição dos custos de produção;
- Aumento da produtividade;
- Resistência a diversas condições (pó, humidade, calor);
- Maior flexibilidade (pode ser aplicada às mais diversas áreas, inclusive a habitações domésticas);
- Melhor qualidade dos processos;
- Maior capacidade tecnológica;
- Solução económica (comparativamente a outros sistemas, p. ex. robótica).

Contudo, possui também algumas desvantagens:

- Como se trata de uma máquina, possui (ainda) capacidade limitada à tomada de decisões;
- Só pode ser programado ou ajustado para controlar uma operação em condições específicas;
- Requer calibração periódica (p. ex. nos temporizadores do programa), de forma a garantir a exactidão do processo;
- Eventualmente necessitará de manutenção, para garantir que a precisão não diminua ao longo do tempo;
- São facilmente susceptíveis a ruídos de equipamentos vizinhos presentes na rede eléctrica.

Existem diversos tipos de automação [8]:

- Automação fixa:
 - ✓ Requer altos investimentos;
 - ✓ Permite a obtenção de altas taxas de produção;
 - ✓ Configuração rígida (difícil de alterar);
 - ✓ Implementa simples operações;
 - ✓ Equipamento específico (p. ex. máquinas para colocação de tampas).

- Automação programável:
 - ✓ Requer também altos investimentos;
 - ✓ Taxas médias de produção (inferiores à automação fixa);
 - ✓ Configuração um pouco mais flexível que a automação fixa (possibilidade de reprogramação);
 - ✓ Equipamento genérico (equipamento adaptável a outra função, apropriado para *batch processing*¹).
- Automação flexível:
 - ✓ Investimento muito elevado;
 - ✓ Taxas médias de produção;
 - ✓ Produção contínua (linhas de montagem, embalagem, etc.);
 - ✓ Configuração flexível (facilidade de alteração por *software*);
 - ✓ Equipamento geral (adequa-se a qualquer tipo de função).

A maior parte dos sistemas modernos de automação a nível de indústria são extremamente complexos e requerem muitos ciclos repetitivos [8].

Um sistema de automação com a mínima complexidade deve possuir os seguintes elementos [8]:

- **Accionamento:** fornece energia ao sistema para atingir um determinado objectivo/acção (p. ex. motores eléctricos, pistões hidráulicos, etc.).
- **Sensorização:** mede o desempenho do sistema ou uma determinada propriedade ou grandeza (p. ex. sensores de temperatura, *encoders*² para medição da velocidade, entre outros).
- **Controlo:** utiliza a informação dos sensores para regular o accionamento.
- **Comparador:** elemento de decisão; compara valores medidos com valores predefinidos (na programação) e toma a decisão de acordo com o que foi estabelecido (novamente, na programação do sistema).
- **Programas:** contêm informações do processo e permitem controlar as interacções entre os diversos componentes do sistema.

¹ *batch processing* - produção por lotes.

² *encoders* - dispositivos electromecânicos que contam ou reproduzem pulsos eléctricos a partir do movimento rotacional dos seus eixos. Podem ser referenciados como transdutores³ de posição angular.

³ *transdutores* - dispositivo que transforma um determinado tipo de energia num outro tipo.

É apresentado na Fig. 1.6 o fluxo de um possível sistema de controlo.

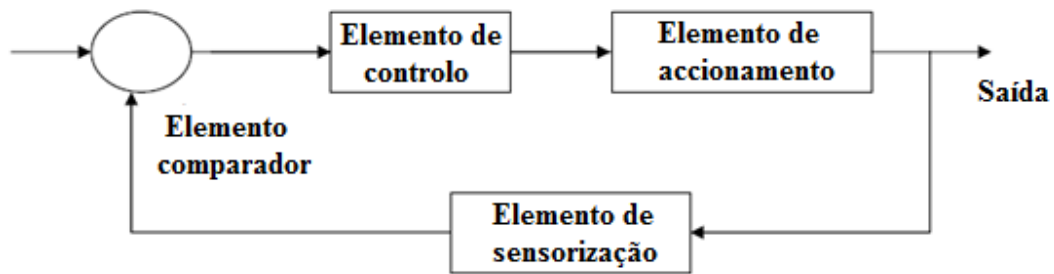


Figura 1.6 - Fluxo de um sistema de controlo [8].

1.1.1. Contextualização

O tema da presente dissertação foi seleccionado devido à actual crise e evolução dos sistemas, pois torna-se cada vez mais apelativo ter equipamentos capazes de transformar processos inicialmente complicados de gerir em processos simples e autónomos (p. ex. automação de uma habitação ou automação de um determinado sistema/processo).

Outro factor que influenciou a escolha do tema foi o facto de, e como os PLC's ocupam grande parte da indústria actualmente, ter a oportunidade de interagir com este tipo de sistemas o quanto antes e adquirir a maior quantidade de informação e experiência possível de forma a alargar o meu conhecimento às mais diversas áreas.

Tendo em conta a minha posição de estagiário à data de realização desta dissertação na empresa TecnoSoares, Lda, surgiu um projecto com o tema adequado à elaboração deste projecto que foi fundamental na escolha e execução do mesmo.

1.1.2. Objectivos

Os objectivos deste trabalho focam-se essencialmente na criação de um sistema de rega inteligente. Este, permitirá o controlo e monitorização de um sistema de abastecimento de água e recolha de dados provenientes dos diversos sensores constituintes do mesmo. Para tal, foi criada uma maquete exemplificativa do sistema final para auxílio à simulação.

Além disso, pretende-se criar uma *interface* táctil através de uma consola HMI, assim como uma página *web*, que permitam a monitorização e controlo quer local quer remoto do sistema.

Pretende-se também criar um sistema capaz de armazenar dados (base de dados) e valores para eventuais comparações e tomadas de decisão.

1.1.3. Organização da dissertação

No primeiro e presente capítulo, são referidos os objectivos do projecto, assim como uma introdução e apresentação dos PLC's, a sua importância e algumas vantagens e desvantagens da sua utilização.

No segundo capítulo segue-se a explicação geral e conceitual dos PLC's bem como um resumo da história dos mesmos. Ainda no mesmo capítulo é realizada uma descrição, explicação dos diversos tipos de PLC's existentes e uma explicação detalhada do funcionamento de um PLC, dos seus módulos e partes constituintes. Termina-se o capítulo com a apresentação das diferentes linguagens de programação de PLC's existentes.

No capítulo seguinte são apresentados e justificados os motivos da utilização do PLC para este projecto; quais as necessidades do projecto e como o PLC e os restantes componentes escolhidos se enquadram nas mesmas. É também apresentado um vislumbre do *software* utilizado para a programação do PLC e respectiva consola HMI, bem como as etapas utilizadas para a realização do projecto. Termina-se o capítulo com a apresentação detalhada do sistema de rega inteligente elaborado para este projecto.

No quarto capítulo são apresentadas as respectivas conclusões à realização desta dissertação, bem como eventuais melhorias futuras a implementar neste sistema.

No quinto e sexto capítulos desta dissertação são apresentadas as devidas referências documentais utilizadas e anexos, respectivamente, referentes à mesma.

2. Capítulo 2

2.1. PLC's

2.1.1. O que é e como funciona um PLC?

Os PLC's, ou *Programmable Logic Controllers* (Controladores Lógicos Programáveis) são dispositivos electrónicos que permitem o controlo de máquinas e processos [9]. Estes, utilizam memórias programáveis para armazenamento de instruções e execução de funções específicas tais como sequenciação, temporização, contagem, cálculos aritméticos, manipulação de informação e comunicação para o controlo de processos [1].

A seguinte figura ilustra um simples diagrama conceitual da aplicação do PLC a um determinado processo.

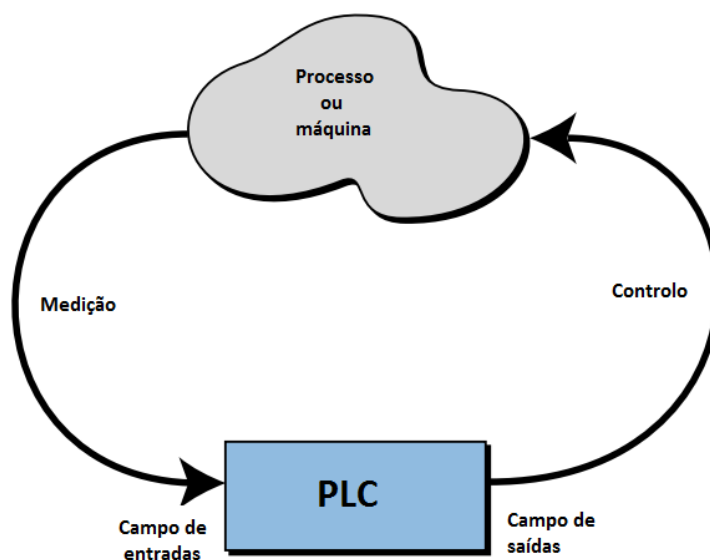


Figura 2.1 - Diagrama da aplicação do PLC a um determinado processo ou máquina [1].

Um PLC consiste nas seguintes secções básicas (Fig. 2.2) [1] [9]:

- A unidade central de processamento ou CPU;
- As *interfaces*⁴ de entradas e saídas;
- Dispositivos de programação (opcional);
- *Interfaces* de operação (opcional).

⁴ *interfaces* - dispositivos físicos ou lógicos que fazem a ligação entre dois tipos de sistemas.

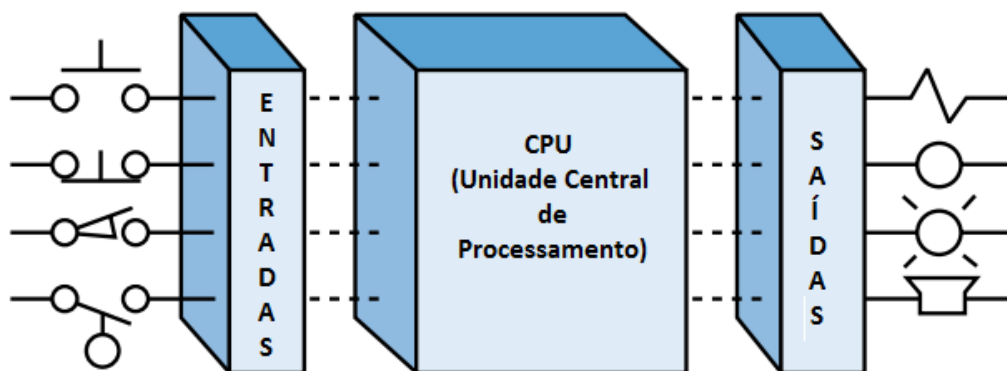


Figura 2.2 - Constituição de um PLC [1].

A CPU tem a seu cargo gerir todas as actividades do PLC. Por sua vez, é composta pelos seguintes componentes (Fig. 2.3) [1]:

- O processador;
- O sistema de memória;
- O sistema de alimentação.

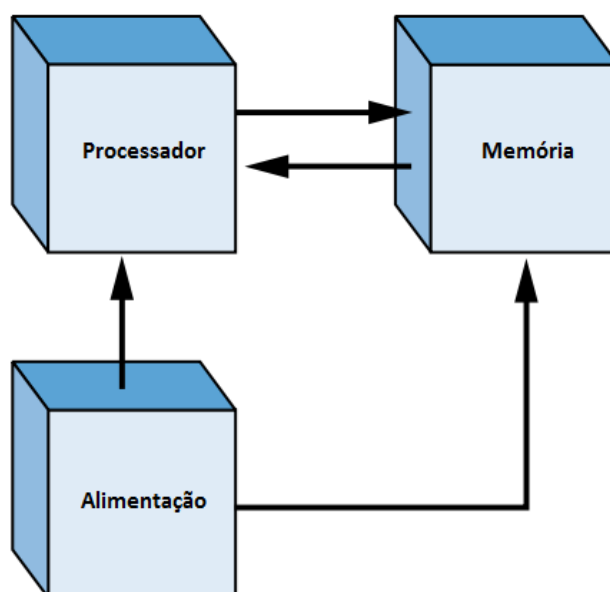


Figura 2.3 - Componentes constituintes da CPU [1].

Cada uma das partes constituintes do PLC e CPU será devidamente apresentada e explicada com maior pormenor na secção “Partes constituintes de um PLC” desta dissertação.

2.1.2. História da automação

Ao longo do tempo, a automação tem vindo cada vez mais a fazer parte integral do quotidiano dos seres humanos. As primeiras iniciativas do homem para automatizar actividades manuais ocorreram na pré-história. Invenções como a roda, o moinho de vento e as rodas de água demonstram a criatividade do homem para poupar esforço e economizar tempo e recursos.

A história da automação industrial começa com a criação das linhas de montagem de automóveis, com Henry Ford, na década de 20 [10]. Durante a década de 50, os dispositivos electromecânicos foram os recursos mais utilizados para efectuar controlos lógicos nas linhas de produção e em máquinas isoladas. Tais dispositivos, baseados essencialmente em relés, apresentavam especial importância na indústria automóvel em que a complexidade dos processos produtivos envolvidos exigia instalações em painéis e cabines de controlo com centenas de relés, e como tal, um ainda maior número de interligações. Tais sistemas de controlo, apesar de funcionais, apresentavam bastantes problemas de ordem prática: quando algum componente falhava, eram necessárias várias horas ou dias para encontrar o problema, essencialmente devido à grande quantidade de elementos de cada sistema. Além disso, o facto dos relés e restantes componentes possuírem grande dimensão física, os painéis de controlo ocupavam bastante espaço e não apresentavam as devidas protecções e imunidades contra a humidade, gases, temperatura, poeira, etc. [11].

Com a evolução da tecnologia, alguns dispositivos baseados em transístores foram utilizados no final da década de 50 e início dos anos 60, sendo que tais dispositivos reduziam muitos dos problemas existentes anteriormente nos relés. Porém, foi com o surgimento dos componentes electrónicos integrados em larga escala que a evolução se tornou mais significativa, especialmente em relação às tecnologias para automação [11].

O primeiro controlador lógico programável surgiu no ano de 1968, na divisão de *hidromática* da *General Motors Corporation* [1]. O principal objectivo era eliminar os elevados custos associados ao inflexível sistema de controlo baseado em relés. Aliado ao uso de dispositivos periféricos capazes de realizar operações de entrada e saída, um minicomputador com a sua capacidade de programação obteve vantagens técnicas de controlo que reduziram o custo de implementação de um sistema daquela dimensão. Surgia assim a era dos controladores lógicos programáveis [11].

2.1.3. Evolução

O primeiro PLC apresentava funcionalidades baseadas em relés, substituindo a original lógica de relés mecânicos, que utilizavam electricidade para operar dispositivos que mecanicamente comutavam os circuitos eléctricos [1]. Estes controladores eram facilmente instalados e ocupavam menos espaço.

A primeira geração de PLC's recebeu melhorias com o avanço da tecnologia dos microprocessadores ocorridos durante os anos 70. Foram adicionados recursos importantes ao longo do tempo, tais como *interfaces* de operação mais simples, instruções de aritmética e manipulação de dados, recursos de comunicação por meio de redes, possibilidade de configurar especificamente cada finalidade com recurso a módulos alteráveis, entre outras [11].

Muitos avanços tecnológicos na indústria de automação continuam a surgir actualmente. Estes avanços não afectam apenas o *design* do PLC, mas também a filosofia do sistema de controlo, o *hardware* (componentes físicos) e o *software* (programa de controlo) [1].

Desde o surgimento dos primeiros PLC's até aos dias de hoje, os avanços mais notórios são baseados nos seguintes pontos [1]:

- Os tempos de *scan* (leitura) são cada vez mais rápidos, graças aos avanços dos processadores e electrónica;
- PLC's mais pequenos, baratos e mais potentes, facilmente substituem 4 a 10 relés;
- Elevada densidade de *inputs/outputs* (I/O) providenciam uma maior eficiência de espaço e baixo custo;
- Processadores e *interfaces* (controladores PID, rede, CANbus, comunicações ASCII, posicionamento GPS, entre outros) de I/O;
- Melhorias no *design* mecânico;
- *Interfaces* especiais vieram permitir a ligação de diversos dispositivos directamente ao PLC, tais como sensores termopar, entradas de comutação rápida, etc.;
- Evolução do equipamento periférico e documentação do sistema faz agora parte integral do sistema.

Todos estes pontos traduzem-se numa enorme evolução no mundo dos PLC's, criando assim PLC's mais potentes, mais sofisticados (> 8000 I/O), reutilizáveis, mais flexíveis, mais confiáveis e eficientes e sobretudo, mais pequenos e seguros.

Actualmente, os PLC's já se apresentam como um sistema de controlo maduro e com mais funcionalidades que os primeiros modelos. São capazes de comunicar entre si, providenciar relatórios, horários de produção e diagnósticos das suas próprias falhas. Os PLC's mais actuais podem ser considerados inovadores em relação ao equipamento que utilizam, e apesar de funções como a operação da velocidade, tipos de *interfaces* e processamento de dados terem evoluído ao longo dos anos, as especificações e princípios ainda se mantêm as mesmas que as dos fabricantes iniciais: serem simples de utilizar e fáceis de manter [1].

Os fabricantes de PLC's entenderam também a necessidade de criar uma *interface* para os sistemas de controlo e passaram a produzir sistemas de controlo e aquisição de dados (*Supervisory Control And Data Acquisition* - SCADA), tendo começado a comercializar o PLC juntamente com o *software* e *hardware* SCADA [10]. Este tipo de *hardware* serve essencialmente para criar ambientes gráficos, controlar e monitorizar ambientes industriais, onde a necessidade de simplificar os processos é elevada, de forma a reduzir a necessidade de formação dos utilizadores/empregados e a complexidade do controlo/monitorização.

O futuro dos PLC's reside não apenas na continuação da evolução em torno de novos produtos, mas também na integração dos PLC's com outros equipamentos e gerenciadores de indústrias [1].

Os novos avanços incluem funcionalidades tais como melhores *interfaces* de operação e controlo, *interfaces* de utilizador gráficas (GUI), módulos de controlo por voz, suporte para ligação a programas e dispositivos de reconhecimento por voz, e inteligência artificial e sistemas de I/O baseados em lógica *fuzzy*⁵.

⁵ *lógica fuzzy* - lógica difusa; extensão da lógica booleana que admite valores intermediários entre o 0 (Falso) e o 1 (Verdadeiro), como por exemplo 0,5 (Talvez).

2.1.4. Partes constituintes de um PLC

Existem vários atributos que influenciam o custo de um PLC, tais como, o tipo de sistema a controlar, o número de entradas e saídas que serão necessárias para o devido controlo e automação do sistema, os módulos de expansão necessários (p. ex. módulos de comunicação GPRS, módulos GPS, módulos de comunicação *wireless*, RS232, entre outros), etc.

Assim, o mercado dos PLC's pode ser segmentado de acordo com o seu tamanho (leia-se número de entradas e saídas) nos seguintes grupos [1]:

- *Micro* - até 32 I/O;
- *Small* - de 32 a 128 I/O;
- *Medium* - de 64 a 1024 I/O;
- *Large* - de 512 a 4096 I/O;
- *Very Large* - de 2048 a 8192 I/O.

Através da análise do número de entradas e saídas de cada categoria, facilmente se chega à conclusão que quanto maior for o número de entradas e saídas, mais elevado será o custo final do sistema.

De seguida serão explicadas detalhadamente as partes constituintes de um PLC.

2.1.4.1. Entradas digitais/discretas

Os terminais de entrada do PLC formam a *interface* através da qual os dispositivos de campo são ligados ao PLC [9]. Em alguns casos, o próprio PLC já possui estas *interfaces* acopladas.

As *interfaces* de entradas podem ser divididas em duas categorias:

- Entradas digitais ou discretas⁷;
- Entradas analógicas.

Às entradas ditas digitais ou discretas só é possível conectar botões, interruptores, sensores digitais/discretos entre outros dispositivos digitais/discretos.

⁷ *discretas* - possuem apenas dois estados: ligado (1) ou desligado (0).

Os PLC's mais antigos estavam limitados a apenas entradas/saídas digitais, fazendo com que o PLC pudesse controlar apenas uma parte de muitos processos, pois muitos deles requeriam a manipulação de valores e medições numéricas para controlar dispositivos analógicos e de instrumentação [1].

Os módulos de I/O assentam na base de conexão do tipo *plug-in*, permitindo assim diversas vantagens e facilidades aquando da montagem do sistema [12]:

- Fácil montagem e desmontagem dos equipamentos;
- Permite a rápida desconexão de um equipamento em caso de disfunção ou avaria;
- Redução até 50% no tamanho da instalação do sistema;
- Aumento da confiança e segurança na conexão dos equipamentos;
- Minimização dos erros de montagem dos equipamentos.

Dentro da categoria de entradas digitais/discretas, esta pode ser dividida nos seguintes grupos [1]:

- Entradas de corrente alternada/corrente contínua - AC/DC;
- Entradas DC;
- Entradas AC/DC isoladas;
- Entradas lógicas de transistor a transistor - TTL;
- Entradas de registo/codificação binária decimal - BCD.

As entradas AC/DC operam de um modo semelhante ao circuito apresentado na Fig. 2.4 e são constituídas por duas partes principais [1]:

- A secção de alimentação;
- A secção lógica.

As duas secções estão normalmente acopladas através de um circuito que as separa electricamente, fornecendo isolamento.

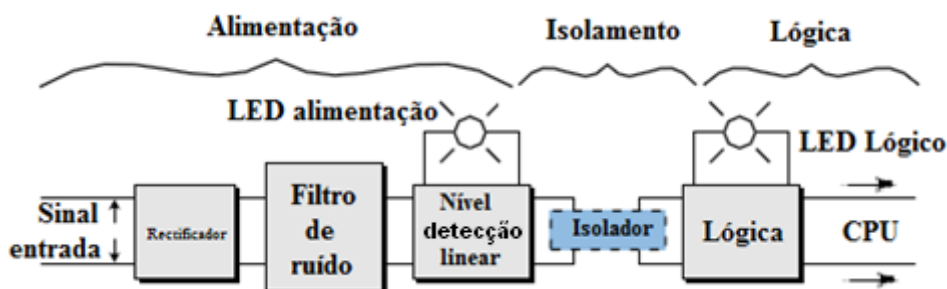


Figura 2.4 - Diagrama de entradas AC/DC [1].

A primeira secção (secção de alimentação) injecta um sinal AC no circuito. Posteriormente, esse sinal vai ser rectificado na ponte rectificadora de díodos, transformando o sinal inicial AC num sinal DC. De seguida, o sinal DC passa pelo filtro com o intuito de remover quaisquer flutuações existentes e ruído eléctrico. Na parte referente ao limiar de detecção, o sinal vai ser analisado da seguinte forma: caso o sinal exceda e se mantenha acima do limiar de tensão durante o tempo de atraso do filtro, o sinal é considerado válido, fornecendo a corrente necessária ao sinalizador do acoplador óptico para acender, dando “ordem” à componente lógica que o sinal de entrada é válido, sinal esse que será posteriormente transmitindo à CPU. Caso o sinal não seja válido, o sinalizador do acoplador não acende e não há passagem de qualquer tipo de informação para a parte lógica do módulo.

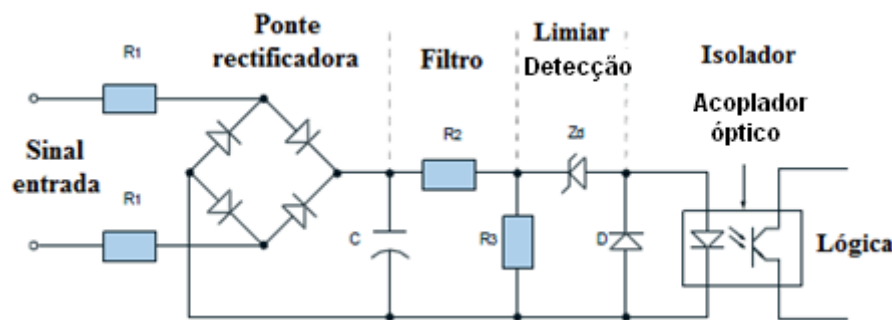


Figura 2.5 - Circuito eléctrico de entradas AC/DC [1].

As entradas DC, como o próprio nome indica, já se encontram na tensão correcta para a transmissão do sinal à componente lógica (tensão DC), não sendo portanto necessário utilizar um rectificador para converter o sinal.

Contudo, os módulos de entradas DC apresentam uma vantagem em relação aos AC/DC pelo facto de conseguirem interagir com os dispositivos de campo em operações de *sinking*⁸ e *sourcing*⁹ [1].

A maior parte dos sensores de proximidade DC utilizados num PLC fornecem uma saída sensorial *sink*, portanto requerem um módulo de entrada *sink* [1].

Na Fig. 2.6 estão representados os dois tipos de funcionamento *sinking* e *sourcing* permitidos nos módulos de entradas respectivos.

⁸ *sinking/sink* - capacidade de um circuito de conduzir corrente em direcção ao potencial ou tensão nulos ou para uma tensão inferior.

⁹ *sourcing/source* - capacidade de um circuito de conduzir corrente de uma fonte de alimentação ou de uma tensão superior no circuito.

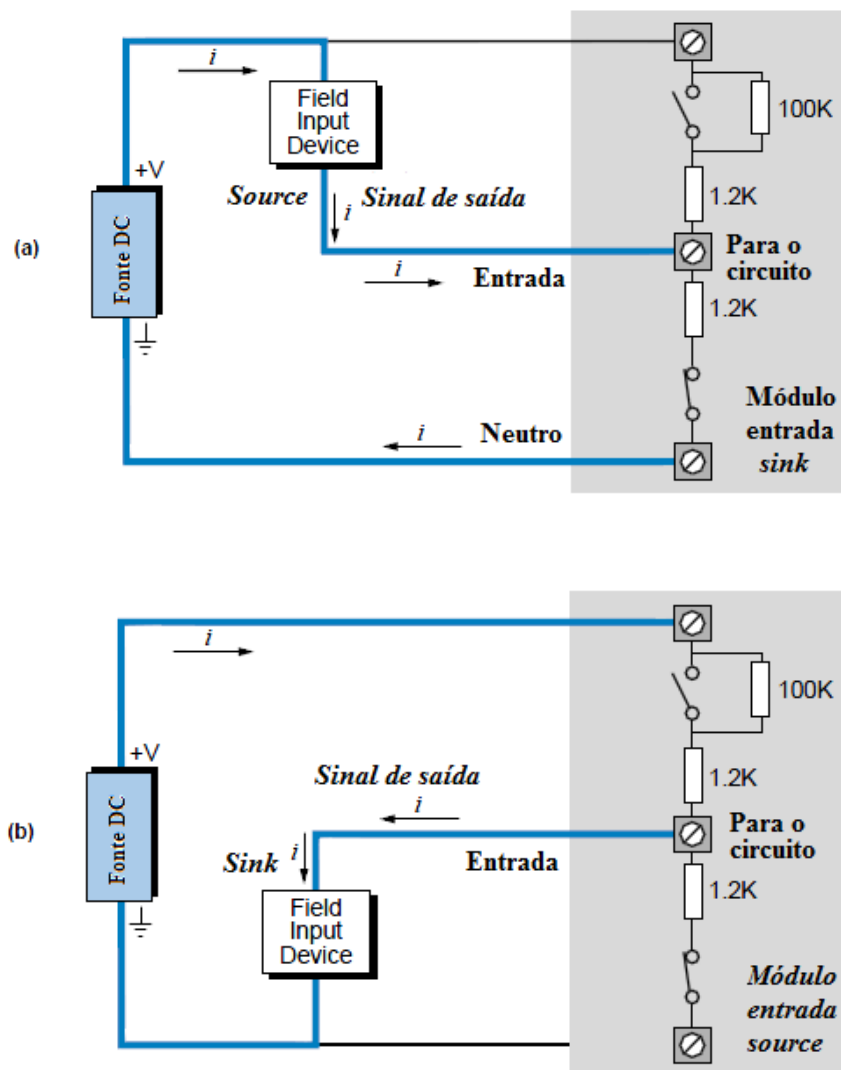


Figura 2.6 - Exemplo de uma entrada *sinking* (a) [1].

Exemplo de uma entrada *sourcing* (b) [1].

As entradas AC/DC isoladas operam de uma forma semelhante aos módulos AC/DC, excepto que cada entrada possui uma linha de retorno separada [1].

Um exemplo deste tipo de aplicação é um conjunto de entradas de dispositivos que estão ligados a diferentes fontes de alimentação. A Fig. 2.7 ilustra uma ligação simples de um módulo de entradas AC/DC isoladas capaz de ligar até 5 dispositivos ao mesmo tempo.

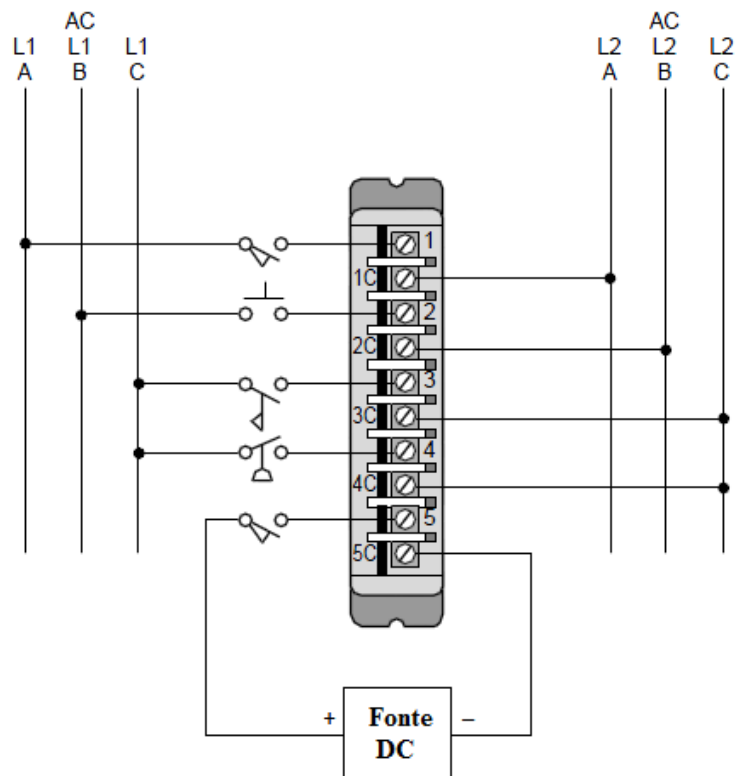


Figura 2.7 - Ligação de um módulo de entradas isoladas AC/DC [1].

As entradas TTL permitem que os controladores aceitem sinais de dispositivos compatíveis com TTL. A sua configuração é bastante similar às entradas AC/DC excepto que o tempo de atraso provocado pelo filtro é bastante menor [1].

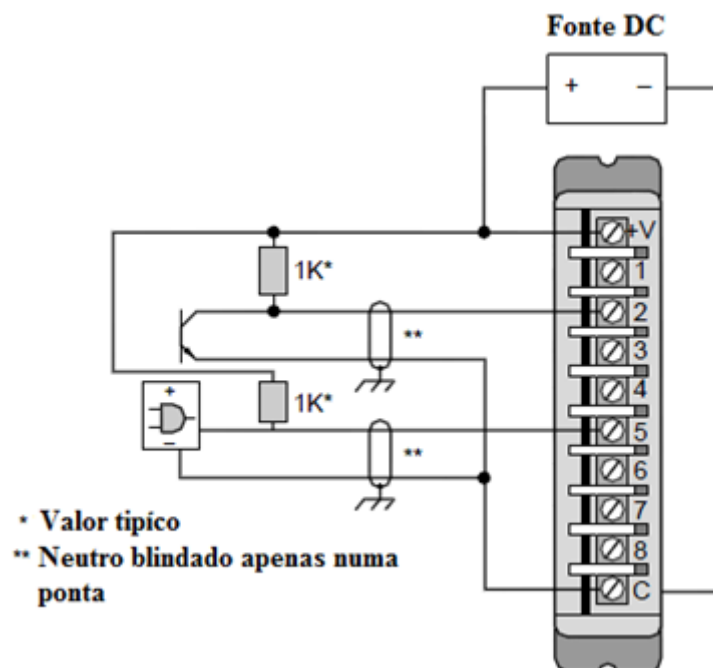


Figura 2.8 - Ligação de um módulo de entradas TTL [1].

Por último, as entradas de registro/BCD permitem que grupos de *bits*¹⁰ sejam introduzidos como uma unidade para acomodar dispositivos que requeiram entrada de *bits* em paralelo [1].

Estas *interfaces* são utilizadas para introduzir parâmetros de controlo do programa em registos específicos ou *word*⁹.

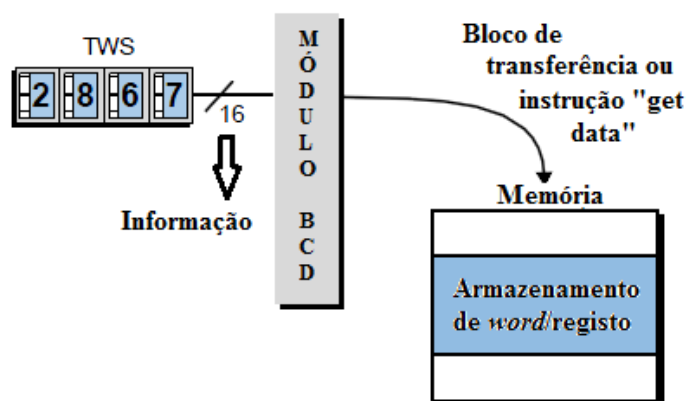


Figura 2.9 - Interface BCD a armazenar dados numa *word*/registo de memória [1].

Muitos fabricantes providenciam capacidades de multiplexagem¹², permitindo assim que mais que uma linha de entrada possa ser conectada a cada terminal num módulo de registo (Fig. 2.10).

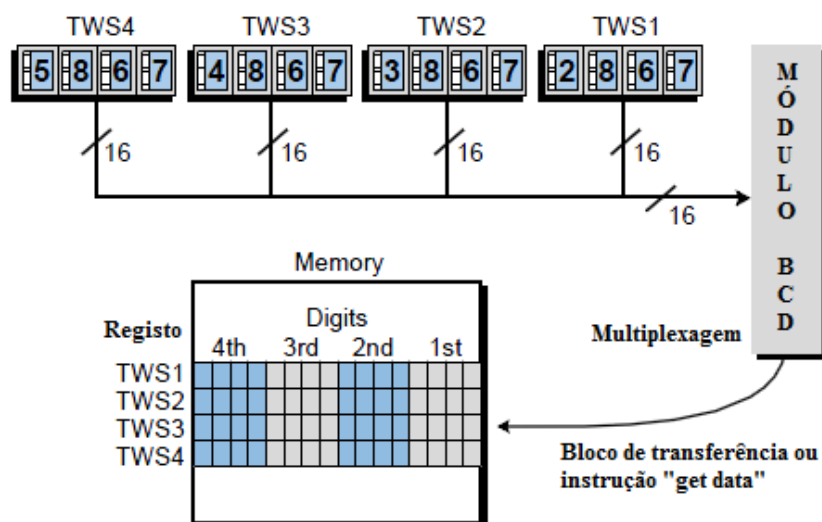


Figura 2.10 - Diagrama exemplificativo da multiplexagem num módulo de entrada BCD [1].

¹⁰ *bits* - unidade de tamanho de memória mais pequena de sistemas digitais, neste caso de um PLC.

¹¹ *words* - espaço de memória com o tamanho total de 16 *bits*.

¹² *multiplexagem* - processo através do qual múltiplos canais de dados, provenientes de diferentes origens, são combinados através de um único canal de transmissão de dados.

2.1.4.2. Entradas analógicas

As entradas analógicas, e como o próprio nome indica, só permitem a ligação de dispositivos analógicos, ou seja, dispositivos que apresentem à sua saída sinais variáveis de corrente e tensão, como por exemplo sensores de pressão e temperatura. Este tipo de entradas veio completar a lacuna existente nas entradas lógicas relativamente ao facto de só aceitarem valores 0 ou 1. Assim, com as entradas analógicas é possível a análise e controlo de dispositivos de sinal contínuo, apresentando um número de estados muito superior ao das entradas digitais (apenas dois estados - 0 e 1). Na Fig. 2.11 está representado um exemplo de um sinal analógico.

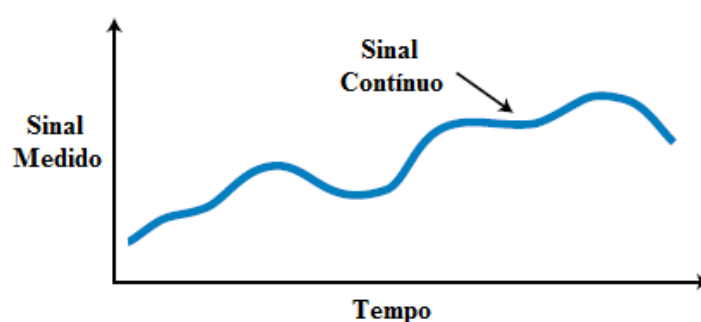


Figura 2.11 - Exemplo de um sinal analógico medido.

Os PLC's, como qualquer outro tipo de computador, apenas interpretam sinais digitais/discretos, ou seja, 0's e 1's. Assim, para conseguirem interpretar os sinais analógicos, estes necessitam de ser convertidos para valores discretos que podem ser interpretados pela CPU.

Alguns exemplos de dispositivos que fornecem sinais analógicos são os seguintes:

- Sensores de humidade;
- Potenciómetros;
- Sensores de temperatura;
- Sensores de vibração;
- Sensores de pressão, entre outros.



Figura 2.12 - Sensores analógicos de humidade (esquerda) [13] e temperatura (direita) [14].

Devido ao elevado número de sensores disponíveis, os módulos de entrada analógicos admitem várias escalas e valores de entrada.

Alguns desses valores mais usuais podem ser encontrados na seguinte tabela:

Tabela 2.1 - Valores típicos admissíveis nas *interfaces* de entradas analógicas [1].

Interfaces de entrada
4 - 20 mA
0 a +1 volts DC
0 a +5 volts DC
0 a +10 volts DC
1 a +5 volts DC
± 5 volts DC
± 10 volts DC

O processo de transformação do sinal analógico para sinal discreto é levado a cabo por um conversor A/D que divide o sinal de entrada em vários sinais digitais. Ao processo de divisão atribui-se o nome de resolução. A resolução do A/D indica em quantas partes o conversor vai dividir o sinal de entrada [15]. No caso de realizar a conversão de um sinal de entrada analógico para um sinal digital utilizando 12 *bits* de um A/D, pode-se dizer que o conversor vai utilizar uma resolução de 12 *bits* ou 4096 partes (pois $2^{12} = 4096$), ou seja, vai dividir o sinal de entrada em 4096 partes. Assim, o sinal inicialmente analógico vai ser convertido para um sinal digital com uma escala entre 0000 e 4095 [1].

Na seguinte figura pode-se visualizar o exemplo de como é efectuada uma conversão recorrendo a um A/D de 3 *bits*.

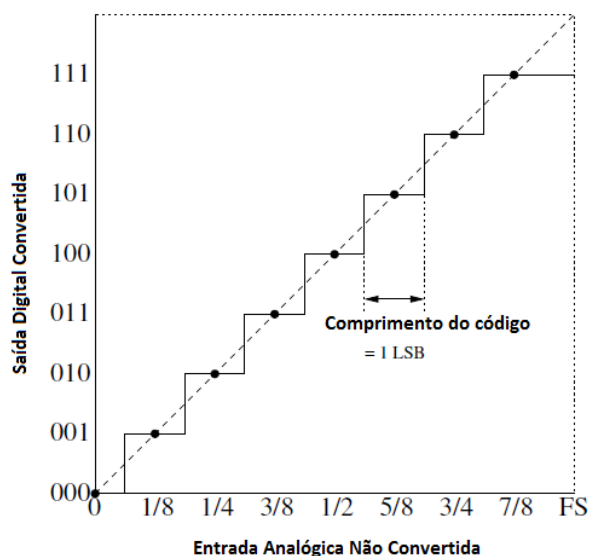


Figura 2.13 - Característica ideal de um conversor A/D com resolução de 3 *bits* [1].

A vantagem da utilização de conversores A/D com maior resolução é que, para a mesma quantidade de informação analógica de entrada, vão haver menos percas de informação, ou seja, enquanto um conversor A/D de 3 *bits* está limitado a dividir o sinal de entrada em apenas 8 partes de informação, num conversor A/D de 12 *bits* esse número de partes já vai aumentar para 4096. Deste modo, conclui-se que é impossível converter a mesma quantidade de informação de um sinal analógico para um sinal digital num conversor de 3 *bits* e num de 12 *bits*, pelo que, alguma informação vai-se “perder” aquando da conversão.

Usualmente, as *interfaces* de entrada analógicas e digitais de um PLC permitem tensões de 220V AC e 24V DC. No entanto, poderá haver outros PLC de outras marcas cujas tensões de entrada admissíveis sejam diferentes.

As *interfaces* de entrada digitais e analógicas recebem esta tensão proveniente de uma fonte de alimentação (no caso de ser 24V DC) ou directamente da rede eléctrica (caso seja 220V AC) e é posteriormente condicionada, para que possa ser utilizada pelo PLC, pois os componentes internos do PLC operam a tensões reduzidas, devendo portanto estar protegidos de eventuais flutuações elevadas de tensão [9].

Estas, apresentam também a vantagem de estarem isoladas fisicamente (isolamento óptico - ver entradas digitais/discretas AC/DC) e a possibilidade de filtrarem os sinais de tensão (conforme observado nas entradas digitais/discretas do tipo AC/DC), de forma a classificá-los como válidos (sinais passíveis de serem interpretados pelo PLC para realizar uma determinada acção) ou não válidos (ruído eléctrico de alta interferência ou estática) [9]. Os filtros de entrada determinam a validade do sinal de acordo com a sua duração. Alguns PLC's têm a vantagem de permitir o ajuste do tempo de resposta dos filtros [9]. Assim, um maior tempo de resposta permite uma melhor filtragem de ruído. No entanto, caso surja a necessidade de operações de alta velocidade (tais como interrupções e contagens), é preferível optar por filtros com menores tempos de resposta.



Figura 2.14 - Exemplos de *interfaces* de expansão de entradas [16].

2.1.4.3. Saídas digitais/discretas

Dispositivos tais como solenóides, relés, contadores, luzes indicadoras, válvulas e alarmes são passíveis de serem ligados às *interfaces* de saída de um PLC. Os circuitos de saída funcionam de maneira similar aos circuitos de entrada mas no sentido inverso, ou seja, enquanto nas entradas os sinais são recebidos pela *interface* e passam por um isolamento óptico¹³ (Fig. 2.15) antes de chegarem à CPU, no caso das saídas, os sinais emitidos pela CPU passam por uma barreira de isolamento (isolamento óptico) antes da corrente chegar aos circuitos de saída [9].

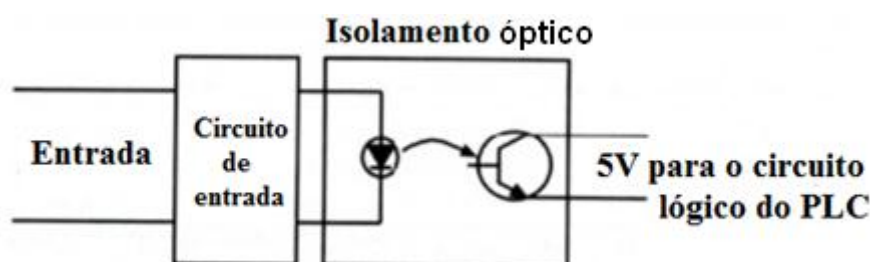


Figura 2.15 - Exemplo de um circuito de isolamento óptico de um PLC [17].

Existem vários tipos de circuitos de *interfaces* de saídas, tais como saídas a relé, a transístores e a triacs [9].

As saídas a relé podem ser utilizadas tanto com corrente AC como DC. A maioria dos relés electromagnéticos presentes nos actuais PLC's aguenta corrente até alguns amperes. A vantagem deste tipo de dispositivos é que suporta melhor eventuais picos de tensão que possam surgir, pois existe uma separação por meio do ar entre as duas extremidades condutoras do relé (o induzido e o núcleo - Ver Fig. 2.16). Contudo, e como a comutação é um processo mecânico, são relativamente lentos quando comparados com os transístores ou triacs e estão sujeitos a um maior desgaste com o tempo de uso.

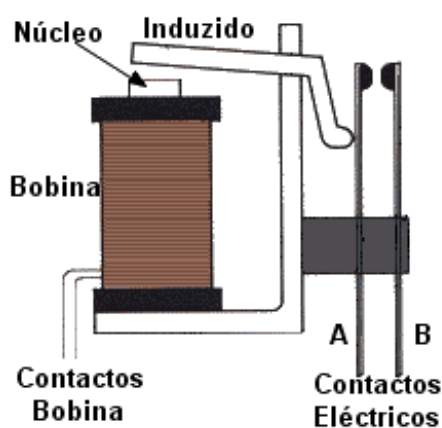


Figura 2.16 - Esquema simples da constituição de um relé [18].

¹³ *isolamento óptico* - isola, no caso dos PLC, o PLC do exterior; não existe nenhuma conexão física entre o PLC e o exterior. Quando é fornecido um sinal quer de entrada quer de saída de um PLC, é ligada uma luz (luz do circuito óptico) que vai iluminar um receptor; quando o receptor recebe luz é activado, permitindo assim a passagem de corrente eléctrica.

Os transístores (Fig. 2.17) por sua vez, operam exclusivamente em corrente contínua. São bastante mais rápidos em termos de comutação que os relés e como não apresentam peças móveis nem nenhum movimento mecânico, são imunes ao desgaste físico. As desvantagens é que suportam apenas correntes no máximo até 0.5A [9]. Alguns tipos de transístores (FET's - Transístores de Efeito de Campo (Fig. 2.18)) podem aceitar cargas superiores, usualmente até 1A.

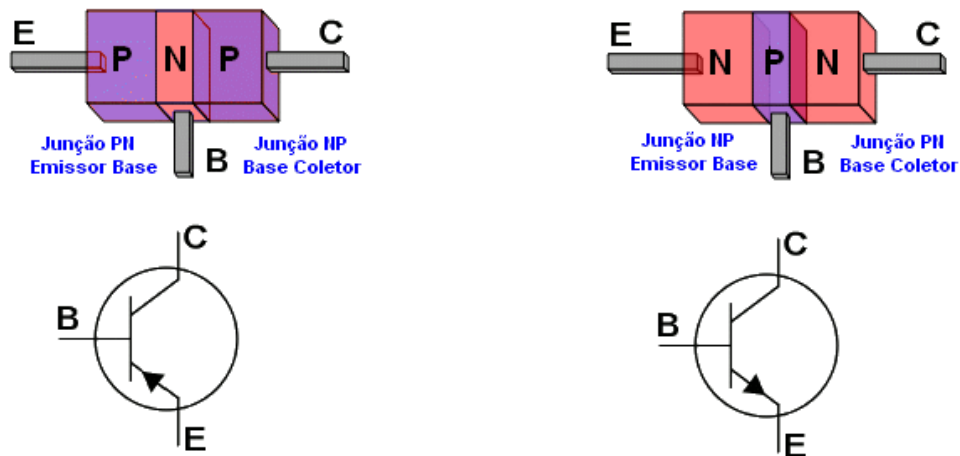


Figura 2.17 - Representação de transístores PNP e NPN e respectivas simbologias [19].

Por último, os triacs possuem características semelhantes aos transístores, excepto que são utilizados para corrente alternada; são de comutação rápida e não estão susceptíveis ao desgaste físico (também por não possuírem peças móveis ou de contacto mecânico). Suportam também cargas de corrente até aos 0.5A [9].

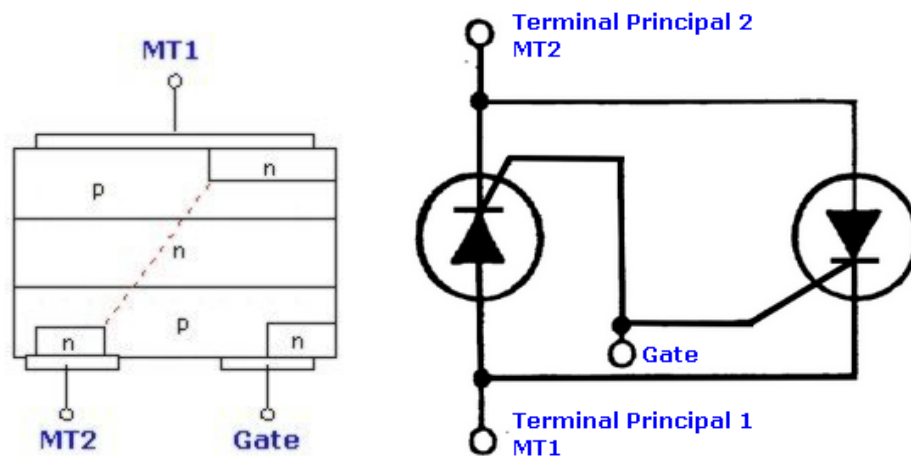


Figura 2.18 - Representação de um triac (esquerda) e a sua simbologia (direita) [20].

Para além das desvantagens enumeradas relativamente aos transístores e triacs face aos relés, em caso de sobrecarga os componentes destes podem ser danificados e/ou destruídos, impossibilitando assim a utilização do endereço de saída do PLC em questão.

Relativamente aos tipos de saídas digitais, e tendo em conta os tipos das entradas, pode-se dizer que são os mesmos, excepto que funcionam em direcção oposta, ou seja, nas entradas digitais o sinal ou informação “viaja” dos dispositivos ligados às entradas, quer analógicas quer digitais, para a CPU. Por sua vez, e como nas saídas o sentido é inverso, o sinal ou informação “viaja” da CPU para os dispositivos ligados quer às saídas analógicas quer às digitais.

Nas tabelas seguintes estão apresentados os tipos de dispositivos passíveis de serem ligados a saídas digitais, bem como os tipos de saídas mais utilizados.

Tabela 2.2 - Dispositivos de saída [1].

Dispositivos de saída
Alarmes
Relés de controlo
Ventoinhas
Buzinas
Luzes
Arrancadores de motores
Solenóides
Válvulas

Tabela 2.3 - Tipos de saída [1].

Tipos de saídas
12 - 48V AC/DC
120V AC/DC
230V AC/DC
Saídas a relé
Saídas TTL
5 - 50V DC (<i>sink/source</i>)

As saídas AC podem variar bastante conforme o fabricante do PLC. Contudo, o seu circuito pode ser representado de acordo com o da seguinte figura.

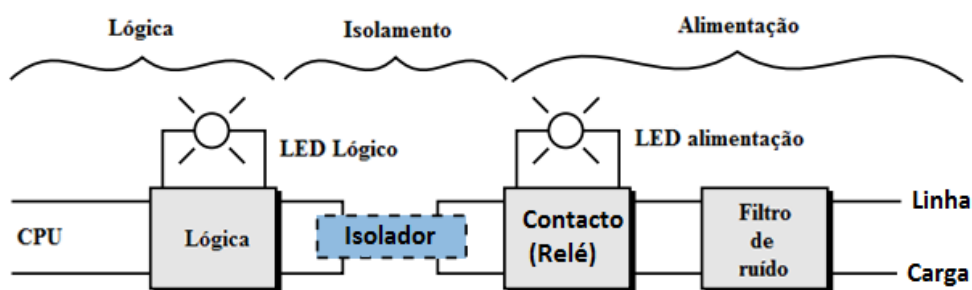


Figura 2.19 - Diagrama de entradas AC [1].

Como se pode observar pela comparação das figuras 2.4 e 2.19, o processo é bastante semelhante, mas no sentido inverso - da CPU para o dispositivo de saída. A única diferença

em termos de componentes físicos do circuito é que as saídas AC não apresentam rectificador, pois caso contrário a tensão que sairia seria DC e não AC como é desejado.

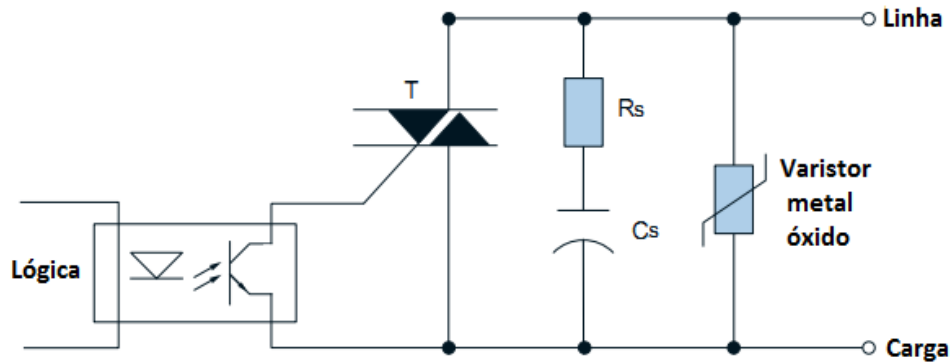


Figura 2.20 - Circuito típico de saídas AC [1].

O circuito de comutação utiliza um triac ou SCR (*Silicon Controlled Rectifier*) para permitir a circulação da alimentação proveniente da carga para a linha (dispositivo de saída). Usualmente, é protegido por um *snubber* RC^{14} ou por um varistor (Metal Oxide Varistor - MOV), limitando a tensão máxima e impedindo que os restantes componentes sejam danificados [1]. Além disso, os *snubbers* e MOV's impedem também a passagem de ruído eléctrico que iria afectar o restante circuito.

As saídas DC funcionam de modo semelhantes às saídas AC, excepto pelo facto de possuírem um transistor de potência (ver Fig. 2.21) para alternar entre a carga e a linha em vez de um relé. Tal como os triacs, os transistores são muito susceptíveis a tensões e correntes elevadas, podendo nesses casos dar origem a situações de curto-circuito (os componentes danificam-se) [1]. Para prevenir essas situações, coloca-se um diodo de roda livre em paralelo com o circuito que vai para o dispositivo de saída. Em alguns casos é também usado um fusível para proteger os transistores durante cargas elevadas, impedindo assim que os transistores sejam danificados [1].

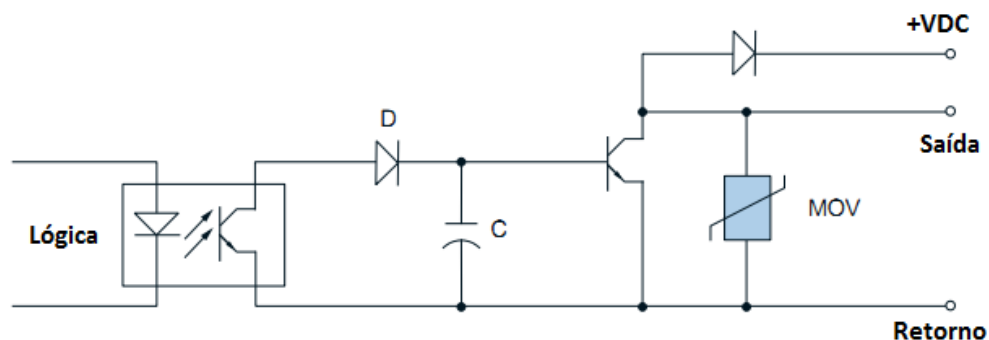


Figura 2.21 - Circuito típico de saídas DC [1].

¹⁴ *snubber* RC - dispositivo (formado neste caso por uma resistência e um condensador em série) utilizado com o intuito de suprimir eventuais variações de tensão em circuitos.

As saídas DC e AC operam de forma semelhante que as saídas AC e DC normais, excepto que cada saída possui uma linha de retorno dedicada isolada das outras saídas [1].

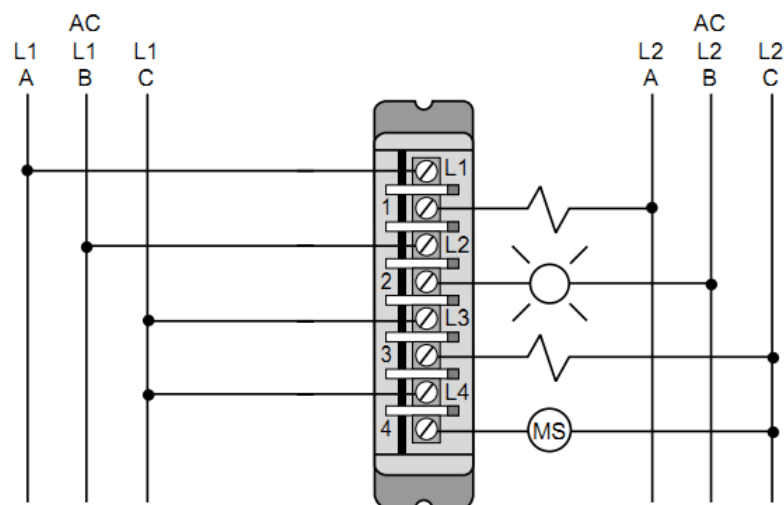


Figura 2.22 - Diagrama de ligação para uma *interface* de saídas AC isoladas [1].

As *interfaces* de saídas TTL permitem que o PLC envie informação para dispositivos que sejam compatíveis com TTL, tais como *displays* de LED's segmentados, circuitos integrados e dispositivos de 5V DC [1]. O seu funcionamento é semelhante aos módulos de entradas TTL, mas, novamente, com a única diferença a residir na direcção do sentido contrário do sinal - da CPU para as saídas.

Tais como as saídas TTL, também as saídas de registo/BCD funcionam da mesma forma que as entradas do mesmo tipo; apenas a direcção do sinal altera (CPU para a saída).

Para terminar a análise das saídas digitais, segue-se uma figura referente ao funcionamento das saídas de registo/BCD.

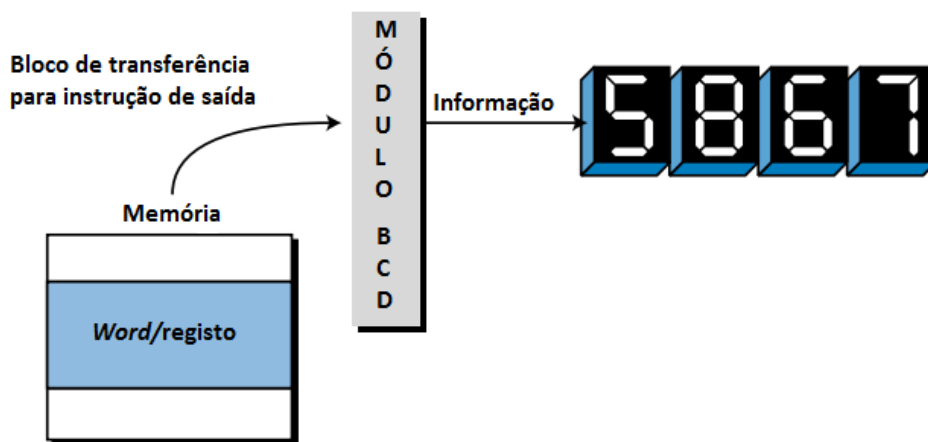


Figura 2.23 - Informação de 16 *bits* a ser enviada para um módulo de saída [1].

2.1.4.4. Saídas analógicas

As saídas analógicas são utilizadas quando surgem aplicações que requeiram o controlo de dispositivos de campo que respondam a variações dos níveis de tensão ou corrente, tais como válvulas analógicas, actuadores, transdutores de pressão, entre outros [1].

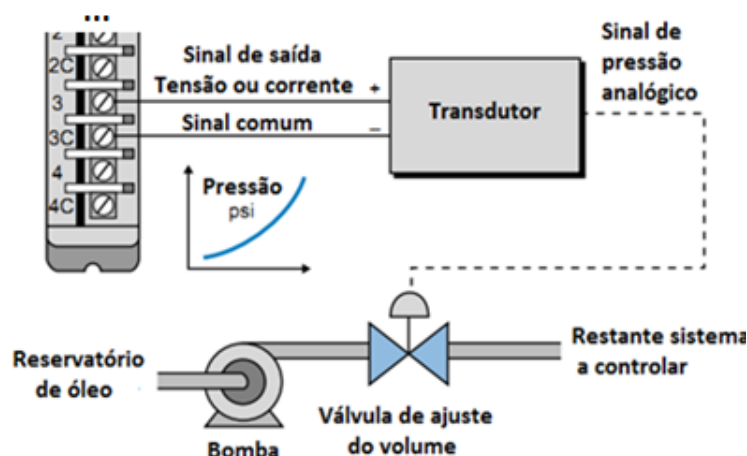


Figura 2.24 - Representação de um circuito capaz de ajustar uma válvula de volume [1].

O circuito da figura 2.24 representa o controlo de uma válvula de volume que por sua vez controla o volume de óleo que é necessário deixar passar para o restante sistema a controlar.

O funcionamento das saídas analógicas é bastante simples (Fig. 2.24): o PLC envia a informação armazenada numa *word* ou registo de memória para o módulo de saídas analógicas; por sua vez o módulo converte (através de um conversor D/A - converte um sinal digital em analógico) essa informação num sinal analógico (tensão ou corrente); posteriormente, esse sinal contínuo é aplicado ao dispositivo ligado à saída do módulo.

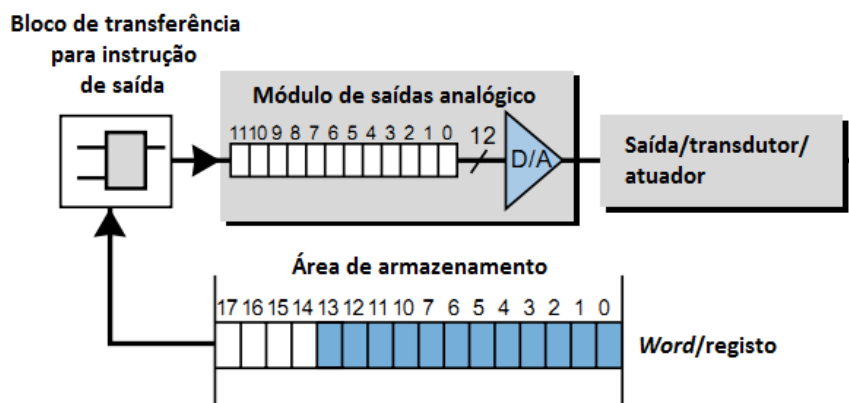


Figura 2.25 - Conversão de um valor digital num valor analógico [1].

Os tipos de saídas analógicas geralmente utilizados são os seguintes:

Tabela 2.4 - Tipos de saídas analógicas [1].

Tipos de saída
4 - 20 mA
10 - 50 mA
0 a +5 volts DC
0 a +10 volts DC
± 2.5 volts DC
± 5 volts DC
± 10 volts DC

2.1.4.5. Processadores (CPU's)

As CPU's dos PLC's são processadores muito pequenos (microprocessadores) sendo considerados o cérebro do PLC e o mais importante componente do mesmo, cuja principal função é “governar” as actividades de todo o sistema [1].

O seu processo de funcionamento é bastante simples: a CPU lê as entradas, executa a lógica segundo as instruções do programa, realiza os cálculos e as conversões necessárias e actua nas saídas [9].

Os utilizadores dos PLC's trabalham com duas áreas distintas da CPU: a área dos arquivos de programa e a área dos arquivos de dados [9].

A área dos arquivos de programas contém o programa de aplicação, as instruções, as sub-rotinas e todas as operações que o PLC deve realizar. Por sua vez, a área dos arquivos de dados armazena dados associados ao programa, tais como, o estado das I/O, valores pré-seleccionados e armazenados em contadores/temporizadores, variáveis, endereços de memória utilizados e outras constantes [9].

Os microprocessadores dos PLC's podem ser categorizados de acordo com o tamanho das suas *words* de armazenamento ou número de *bits* que utilizam simultaneamente para a realização das operações [1].

Relativamente ao tamanho das *words*, estas podem ter 8, 16 ou 32 *bits* [1]. O tamanho das *words* afecta a velocidade com que o processador realiza as operações (quanto maior for o tamanho das *words* mais rápido é o processador - 32 *bits* > 16 *bits*). Ainda dentro da CPU, encontra-se um programa executável ou memória do sistema que direcciona e realiza as actividades de operação, tais como a execução do programa do utilizador, a coordenação da leitura das entradas e actualização das saídas [9]. Este programa executável está constantemente a ler, escrever, realizar operações, conversões e a activar as saídas conforme

especificado no código do programa. A este processo de constante e permanente funcionamento dá-se o nome de *scan*.

Na figura seguinte está apresentado um esquema exemplificativo deste processo.

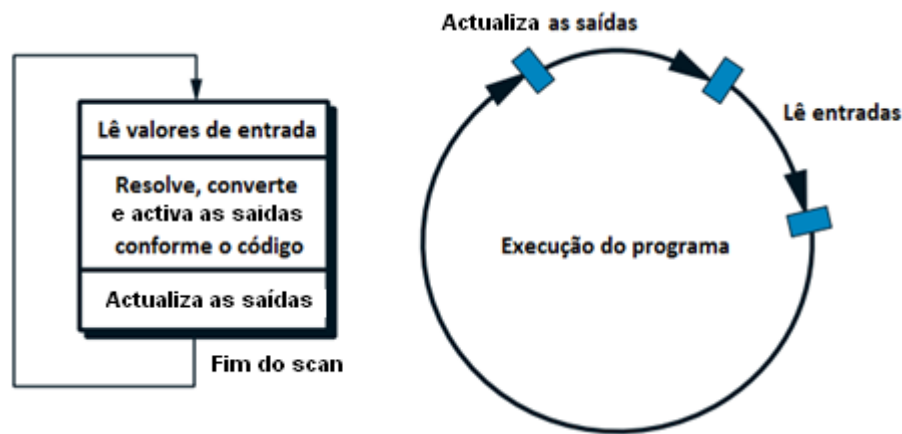


Figura 2.26 - Representação do *scan* de um PLC [1].

Outro factor importante é designado de tempo de *scan* (*scan time*). O tempo de *scan* é o tempo total que o PLC demora a completar um *scan* completo e depende de dois factores: da quantidade de memória utilizada pelo código do programa e do tipo de instruções utilizadas no programa [1]. Nos autómatos mais recentes já é possível ajustar o tempo de *scan* do PLC, como é o caso do autómato utilizado neste projecto (Siemens S7-1200).

O *scan* é um processo contínuo e sequencial de leitura dos estados das entradas que avalia o controlo lógico e actualiza as saídas [1]. Contudo, é necessário ter atenção para não utilizar o PLC para ler alterações nas entradas que ocorram mais rapidamente que o tempo total de um *scan* completo (p. ex. uma entrada que varia em 3 ms e a CPU tem um tempo de *scan* de 10 ms). Caso tal aconteça, possivelmente o programa não irá funcionar de acordo com o esperado (Fig. 2.27).

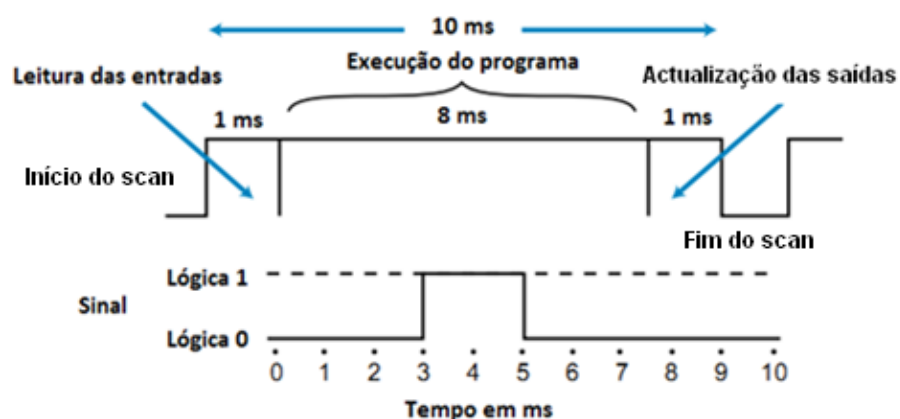


Figura 2.27 - Exemplo de um sinal de entrada que não será detectado pelo PLC [1].

2.1.4.6. Memória

A memória é um dos componentes mais importantes na constituição de um PLC quando se pretende alterar o programa de controlo de forma rápida e simples. O sistema de memória é a área da CPU onde todas as sequências de instruções ou programas são armazenadas e executadas pelo processador. A secção de memória que contém os programas de controlo pode ser alterada ou reprogramada para se adaptar aos requerimentos do processo a automatizar.

O sistema de memória de um PLC é composto por duas secções de memória [1]:

- Memória executiva;
- Memória de aplicação.

A memória executiva consiste num conjunto de programas armazenados permanentemente e que são considerados parte integral do PLC. Estes programas, designados de supervisores, controlam todas as actividades do sistema, tais como a execução do programa de controlo e as comunicações com os restantes dispositivos. Este tipo de memória não permite o acesso do utilizador, pois é onde se encontram as informações referentes às instruções do sistema, como por exemplo, as instruções de relés, as funções de blocos de transferência, instruções matemáticas, entre outras [1].

Por sua vez, a memória de aplicação providencia armazenamento para as instruções e programas criados pelo utilizador [1]. Esta secção de memória é composta por várias áreas (que serão devidamente explicadas mais à frente), cada um delas com a sua função e utilização específicas.

Além das duas secções de memória existentes, as memórias podem ainda ser catalogadas de acordo com o seu tipo:

- **Memória volátil:** este tipo de memória perde o seu conteúdo se toda a alimentação for perdida ou removida, quer através da alimentação normal quer da alimentação de apoio [1]. A memória volátil é facilmente alterada e adequada para a maioria das aplicações quando suportadas por uma bateria de apoio ou a possibilidade de armazenar o programa num disco.
- **Memória não volátil:** pela mesma lógica mencionada acima, este tipo de memória retém a programação e o seu conteúdo mesmo após perda total de alimentação (não necessitando por isso de uma fonte de alimentação de apoio). Este tipo de memória é geralmente inalterável. Contudo, existem alguns tipos de memória não voláteis passíveis de serem alterados [1].

Hoje em dia, os PLC's já oferecem os dois tipos de memória individualmente, podendo em alguns casos, estarem ambos presentes.



Figura 2.28 - Exemplo de um *chip* de memória para um PLC [21].

Existem duas grandes preocupações relativamente ao tipo de memória onde o programa do utilizador está armazenado. Como a memória é responsável pelo armazenamento do programa, a volatilidade é a principal preocupação, pois caso ocorra uma falha de energia, o programa desaparecerá e ocorrerão atrasos ou falhas na produção. Outro factor preocupante deve ser a facilidade com que o conteúdo da memória é alterado [1].

De seguida são apresentados os tipos de memória e em que sentido as suas características afectam a forma como os programas são armazenados, retidos e alterados.

Memória ROM: desenhada para armazenar permanentemente um programa fixo e não alterável em circunstância normais. O seu nome deriva do facto de esta memória ser passível de ser lida mas não alterada (*Read-Only* que significa apenas de leitura). Normalmente, este tipo de memórias são imunes à alteração do seu conteúdo devido ao ruído eléctrico ou perda de alimentação. Os programas executivos (memória executiva) são comumente armazenados na ROM [1].



Figura 2.29 - Exemplo de memórias ROM [22].

Memória RAM: muitas vezes designada de memória R/W (memória de leitura/escrita), foi desenhada para permitir ser lida e escrita [1]. Ao contrário da memória ROM, não retém os seus dados caso haja uma falha de alimentação, sendo portanto considerada uma memória do tipo volátil.

Uma das vantagens deste tipo de memória é a facilidade com que se criam/alteram programas. Além disso, é uma memória de acesso relativamente rápido [1]. A única desvantagem deste tipo de memória é a sua já referida necessidade de estar ligada a uma fonte de alimentação de apoio, e eventualmente caso a alimentação de apoio falhe, os dados serão perdidos.



Figura 2.30 - Exemplo de memória RAM [23].

Memória PROM: tipo especial de memória ROM, pois é passível de ser programada. Poucos PLC's utilizam hoje em dia este tipo de memória e quando utilizada, serve como memória do tipo RAM para algum tipo de *backup* permanente de dados. Assim, este tipo de memória apresenta a vantagem de ser não volátil. Contudo, é relativamente difícil de modificar e requer equipamento de programação especial para tal [1].



Figura 2.31 - Exemplo de memória PROM [24].

Memória EPROM: tipo de memória PROM desenhada para poder ser programada após ter sido completamente apagada por uma luz ultravioleta. Para tal, a janela do *chip* (ver fig. 2.32) deve ser totalmente exposta a uma luz ultravioleta durante aproximadamente 20 minutos [1]. As memórias EPROM providenciam um excelente dispositivo médio para programas que requeiram não volatilidade, mas que não requeiram alterações de dados *online*.



Figura 2.32 - Exemplo de memória EPROM [11].

Uma aplicação de memória composta totalmente por uma memória EPROM é inadequado para alterações *online* ou entrada de dados [1]. Contudo, muitos PLC's possuem um apoio opcional para RAM passível de ser alimentado por uma bateria. Assim, a EPROM armazena os dados permanentes, enquanto a RAM, que é facilmente alterada, torna o sistema apto a ser facilmente modificado.

Memória EAROM: tipo de memória similar à EPROM, mas requer apenas a aplicação de uma determinada tensão nos pinos correspondentes para se proceder à remoção completa dos dados no seu interior, ao contrário de uma luz ultravioleta no caso das EPROM's. Muito poucos PLC's utilizam este tipo de memória, apesar de ser um tipo de memória não volátil e capaz de permitir o armazenamento de dados permanente [1].

Memória EEPROM: circuito integrado de armazenamento que foi desenvolvido por volta de 1970. Como as memórias ROM e EPROM, é um tipo de memória não volátil e oferece ainda a mesma facilidade e flexibilidade de programação que a memória RAM. Tal, faz com que seja um dos tipos de memórias mais utilizados actualmente nos PLC's. Uma desvantagem deste tipo de memória reside no facto de ser escrita apenas para um *byte*¹⁵ de memória após este ter sido previamente apagado, originando assim um atraso. Este período de atraso é perceptível quando são feitas alterações ao programa *online*. Outra desvantagem da EEPROM é a limitação do número de vezes que um *byte* de memória pode ser escrito/apagado. Contudo, quando comparada com os restantes tipos de memórias, as vantagens superam em larga escala as desvantagens tornando-as facilmente desprezáveis [1].

¹⁵ *byte* - conjunto de 8 *bits*.

As memórias dos PLC's podem ser vistas como grandes matrizes bidimensionais de um único bloco de células de armazenamento, cada uma contendo um pedaço de informação na forma de 0 ou 1 - código binário. Dado que cada célula pode armazenar apenas um dígito binário, é designada de *bit* [1]. Como já foi referido anteriormente, um *bit* é a unidade estrutural de memória mais pequena que existe. Apesar de cada *bit* poder armazenar informação na forma de 0's ou 1's, as células de memória não contêm os números 0 nem 1, mas sim cargas de tensão, ou seja, um determinado valor de tensão representa o estado lógico 1 e a ausência de tensão representa o estado 0. Um *bit* é considerado ligado se a informação for 1 (presença de tensão) e desligado se a informação for 0 (ausência de tensão) [1].

Normalmente, as CPU's processam e armazenam os dados em grupos de 16 *bits*, também designados de *words* [9]. Contudo, os utilizadores conseguem manipular os dados ao nível dos *bits*.

Cada *word* de dados possui uma localização específica na CPU designada de endereço ou registo onde cada elemento do programa do utilizador é referenciado com um endereço para indicar onde se localizam os dados para o elemento em questão. Ao atribuir endereços às I/O de um programa, esses endereços estarão relacionados com o terminal de I/O onde os dispositivos de entrada e saída estão ligados [9]. A figura seguinte ilustra a unidade estrutural típica de uma *word* de memória.

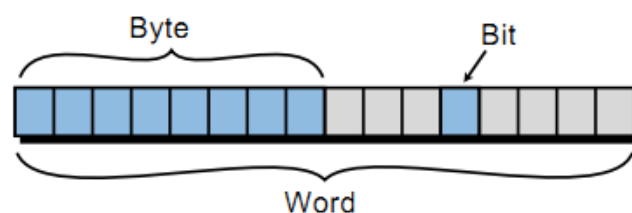


Figura 2.33 - Estrutura de uma *word* de memória [1].

Como se pode observar pela figura 2.33, e como referido acima, um *bit* representa a unidade básica mais pequena de informação. Um *byte* por sua vez é um conjunto de 8 *bits* e uma *word* é constituída por 16 *bits*.

A capacidade de memória de um PLC é vital quando se tem em conta o tamanho de um programa, pois quanto maior for a quantidade de memória de um PLC, maior será o seu custo final. Em pequenos PLC's, a quantidade de memória não é passível de ser aumentada (regra geral em PLC's com menos de 64 I/O). Por sua vez, em autómatos superiores, essa quantidade de memória pode ser expandida [1].

Os tamanhos dos módulos de memória são especificados em termos de unidades K, onde cada unidade K representa 1024 *words*. Assim, uma memória de 1K contém capacidade para armazenar 1024 *words*, 2K contém 2048 endereços para *words*, 4K contém 4096 endereços, e assim sucessivamente [1].

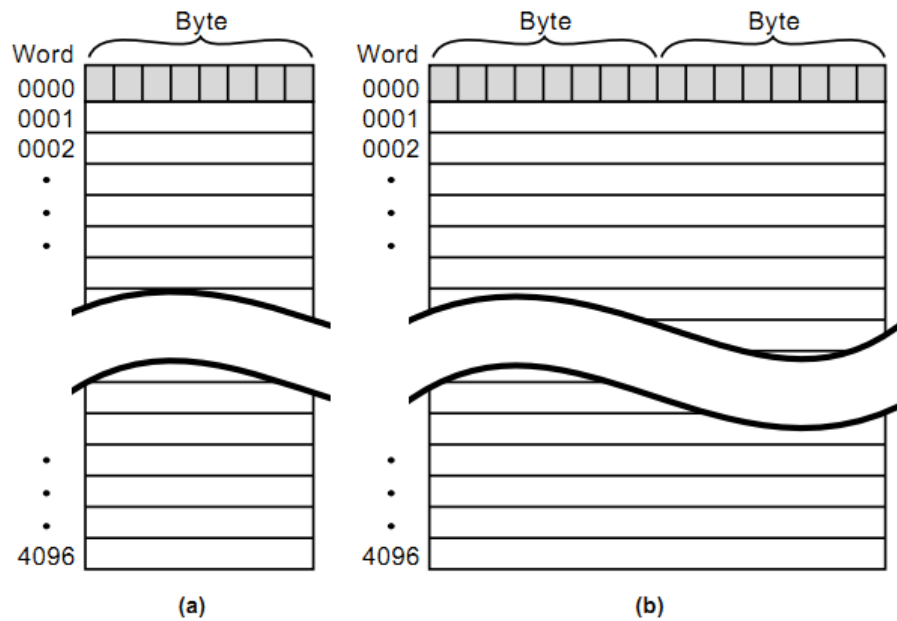


Figura 2.34 - Ilustração de um bloco de 4K com armazenamento de 8 *bits* (a) [1].
Ilustração de um bloco de 4K com um armazenamento de 16 *bits* (b) [1].

A organização da memória, como referido anteriormente, é constituída por duas secções (Fig. 2.35): a memória do sistema ou memória executiva e a memória das aplicações [1].

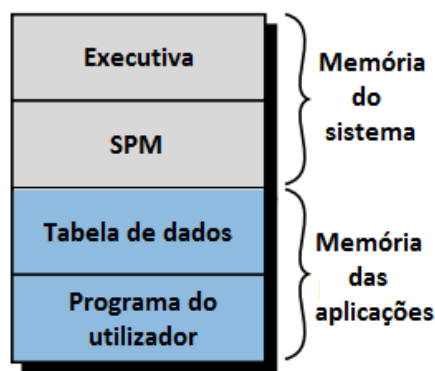


Figura 2.35 - Mapa de memória simplificado [1].

De um modo geral, todos os PLC's devem ter memória alocada em 4 áreas básicas, sendo elas:

- **Área executiva:** área da memória onde estão armazenados permanentemente os programas que são considerados parte do sistema. São sobretudo programas supervisores de actividades tais como execução de programas de controlo, comunicação com dispositivos e actividades normais da CPU [1].

- **Área Scratch Pad (SPM):** área de armazenamento temporária utilizada pela CPU para armazenar pequenas quantidades de dados para cálculos e controlo. A CPU armazena rapidamente os dados que necessita nesta área de memória para evitar longos tempos de acesso relativos à pesquisa de dados na memória principal [1].
- **Área da tabela de dados:** esta área armazena todos os dados relativos ao programa de controlo, tais como valores de temporizadores, de contadores e outras constantes e variáveis utilizadas pelo programa ou CPU. Retém informação acerca de ambas entradas e saídas do sistema. [1].
- **Área do programa do utilizador:** como o próprio nome indica, esta área providencia armazenamento para instruções programadas pelo utilizador. Armazena também o programa de controlo [1].

Como as áreas executiva e SPM estão armazenadas na memória executiva, não podem ser alteradas pelo utilizador. Por outro lado, as áreas de tabela de dados e programa do utilizador já se encontram na memória das aplicações, pelo que podem ser alteradas pelo utilizador.

Relativamente à informação das entradas e saídas, estas estão armazenadas na área de tabela de dados, pertencente à memória das aplicações.

Na seguinte figura está representado um esquema mais detalhado da secção de memória em questão.

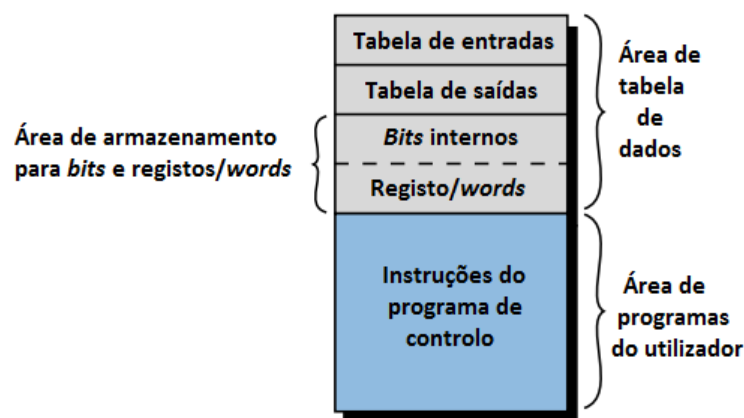


Figura 2.36 - Mapa da memória de aplicações [1].

Por sua vez, a secção de tabela de dados é subdividida em várias áreas:

- Tabela de entradas;
- Tabela de saídas;
- Área de armazenamento.

Tabela de entradas: a tabela de entradas é uma matriz de *bits* que armazena o estado digital das entradas ligadas às *interfaces* de entradas do PLC [1]. O número máximo de *bits* de entrada é igual ao número máximo de campos de entrada que podem ser ligados ao PLC, por exemplo, um PLC com 64 entradas requer uma tabela de entradas de 64 *bits* [1].

Durante a operação do PLC, o processador vai ler o estado de cada entrada correspondente no módulo de entradas e colocar um valor (0 ou 1) no endereço correspondente na tabela de entradas [1]. Os estados das entradas estão constantemente a ser actualizados de acordo com a actualização de I/O.

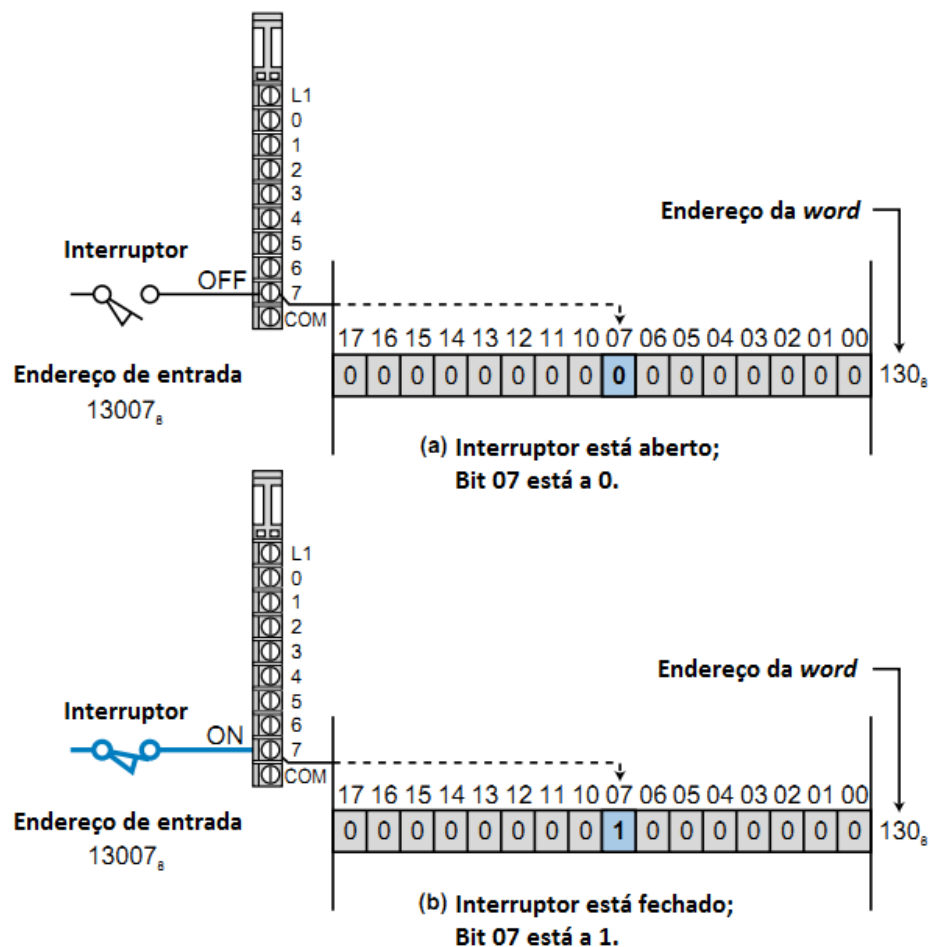


Figura 2.37 - Alteração de um *bit* na tabela de entradas [1].

Tabela de saídas: da mesma forma que a tabela de entradas, a tabela de saídas é uma matriz de *bits* que controla o estado de saída de dispositivos de saída digitais que estão ligados à *interface* de saídas do PLC [1]. Novamente, o número máximo de *bits* disponíveis corresponde ao número máximo de saídas existentes nos módulos de saída do PLC. Quando a CPU dá ordem a um determinado *bit* para ficar a 0, essa informação é transmitida à tabela de saídas que por sua vez actualiza o estado do endereço correspondente ao módulo de saídas. Da

mesma forma que quando é enviado um sinal para colocar um *bit* a 1, o estado do *bit* da saída correspondente vai ser actualizado para 1. A actualização do estado das saídas é feita após o fim de cada ciclo de *scan*.

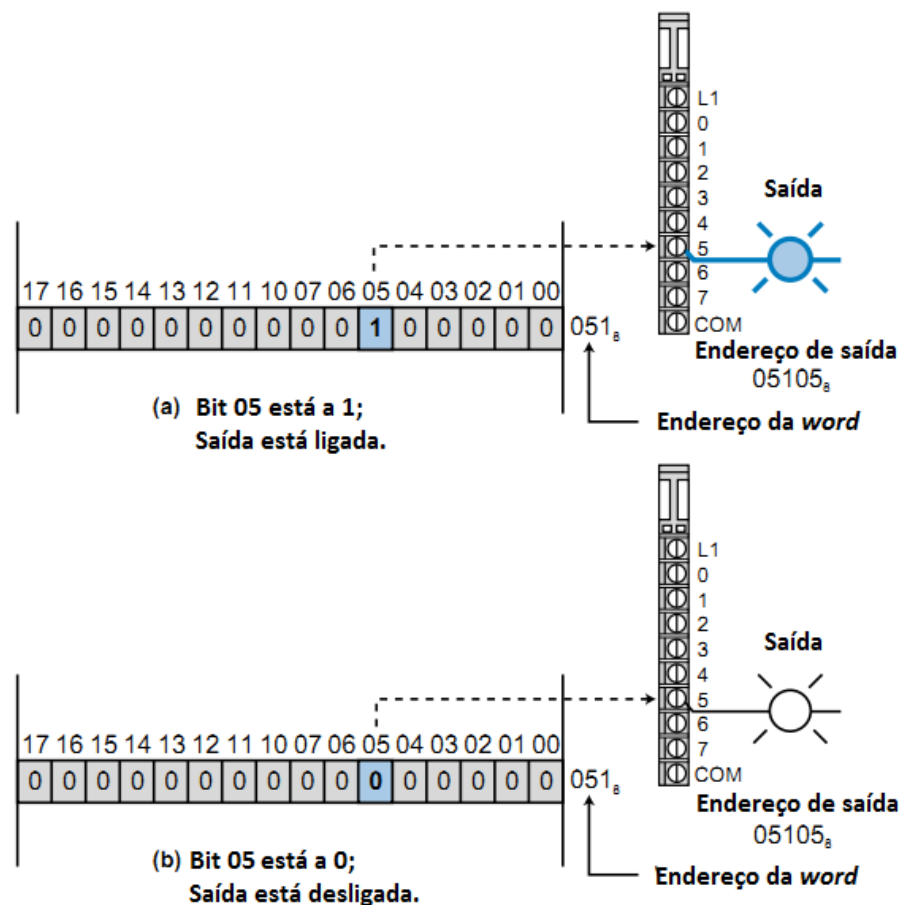


Figura 2.38 - Alteração de um *bit* na tabela de saídas [1].

Área de armazenamento: o propósito da secção da área de armazenamento da tabela de dados é armazenar os dados alteráveis, quer seja um *bit* ou uma *word*. O armazenamento consiste em duas partes: uma área de *bit* interna de armazenamento e uma área de armazenamento de registo/*word*. A área de *bit* interna contém *bits* de armazenamento que são designados de diversas formas: saídas internas, bobinas internas, relés de controlo internos ou somente internos. Estes providenciam uma saída de sequências *ladder*¹⁶ no programa de controlo. Saídas internas não controlam directamente dispositivos de saída pois estão armazenados em endereços que não mapeiam a tabela de saída, e portanto, não afectam os dispositivos de saída [1].

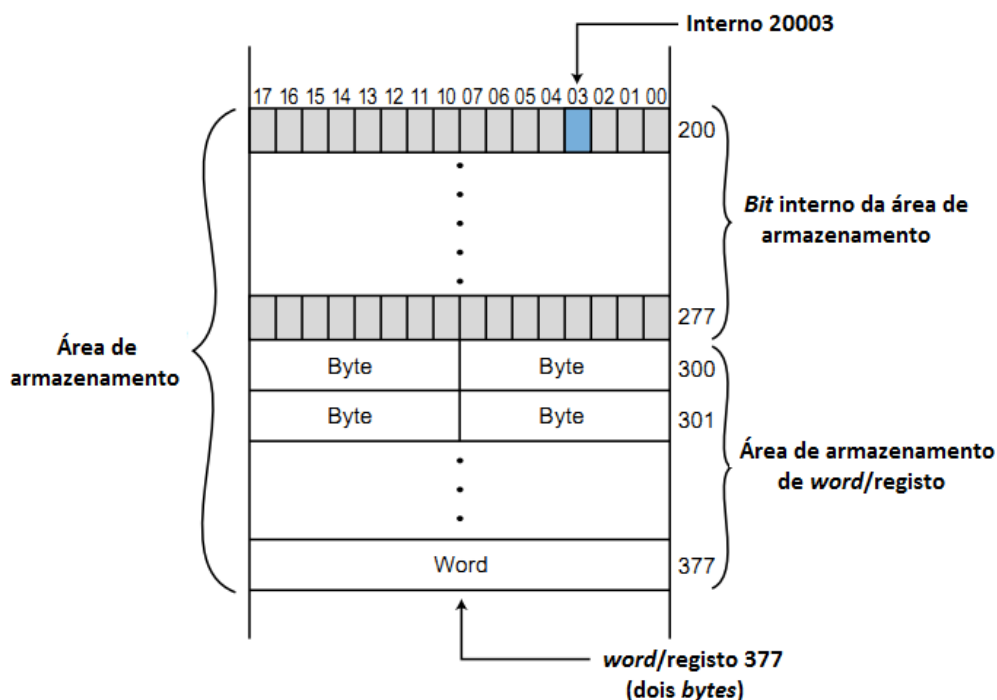


Figura 2.39 - Área de armazenamento da tabela de dados [1].

Os valores introduzidos na área de armazenamento de registo/*word* representam os dados de entrada de uma variedade de dispositivos, tais como interruptores, entradas analógicas e outro tipo de variáveis. Além dos valores de entrada, os registos podem obter valores de saída que são destinados às *interfaces* de saída conectadas a dispositivos, tais como medidores analógicos, *displays* de LED, válvulas de controlo e controladores de velocidade. Os registos de armazenamento são também utilizados para constantes fixas, tais como valores de temporizadores, contadores e resultados de funções aritméticas [1].

2.1.4.7. Alimentação

A fonte de alimentação fornece energia aos elementos electrónicos internos do controlador, converte a tensão de entrada numa tensão adequada à alimentação do PLC e protege os componentes do mesmo contra eventuais picos de tensão [9].

Usualmente, os PLC's requerem alimentação de uma fonte AC, embora alguns PLC's aceitem fontes de alimentação DC o que permite uma maior escolha e diversidade na altura de comprar um PLC de acordo com as necessidades de cada um.

¹⁶ *ladder* - tipo de linguagem utilizado na programação de PLC's.

A maior parte das instalações possui flutuações de tensão na linha e portanto, as fontes de alimentação são projectadas para manter a operação normal mesmo quando a tensão apresenta variações entre 10 e 15% [9]. Quando a tensão de entrada excede o valor máximo ou mínimo de tolerância durante uma duração pré-determinada, a maior parte das fontes de alimentação envia um comando ao processador para desligar, evitando assim eventuais danos nos componentes [1].

Em condições particularmente instáveis de tensão, é preferível a colocação de um estabilizador de tensão entre o PLC e a fonte primária de alimentação [9].

As fontes de alimentação de melhor qualidade possuem boas tolerâncias em flutuações normais de linha, mas nem as melhores construídas e desenhadas são capazes de compensar a especial instabilidade presente em alguns ambientes industriais. Algumas condições causadoras destas instabilidades são [1]:

- Paragem/arranque de maquinaria pesada, tais como motores, bombas, compressores e sistemas de ar condicionado;
- Perdas de linha naturais que variam conforme a distância entre as subestações;
- Perdas do sistema de alimentação causadas por más ligações internas;
- Situações em que a tensão de linha é intencionalmente reduzida pela companhia concessionária.

Um transformador de tensão constante compensa as alterações de tensão ocorridas na entrada (primário) para manter uma tensão constante na saída (secundário). A percentagem de regulação varia conforme a função de operação da carga (alimentação do PLC e dispositivos de entrada) - quanto maior for a carga, maiores serão as flutuações [1].

A razão do transformador de tensão constante (em unidades de volt-ampere - VA) deve ser seleccionada consoante o pior caso de requerimentos de alimentação da carga. O valor desta razão pode ser obtido através do construtor do PLC [1].

A figura seguinte ilustra uma ligação simplificada entre um transformador de tensão constante e um PLC.

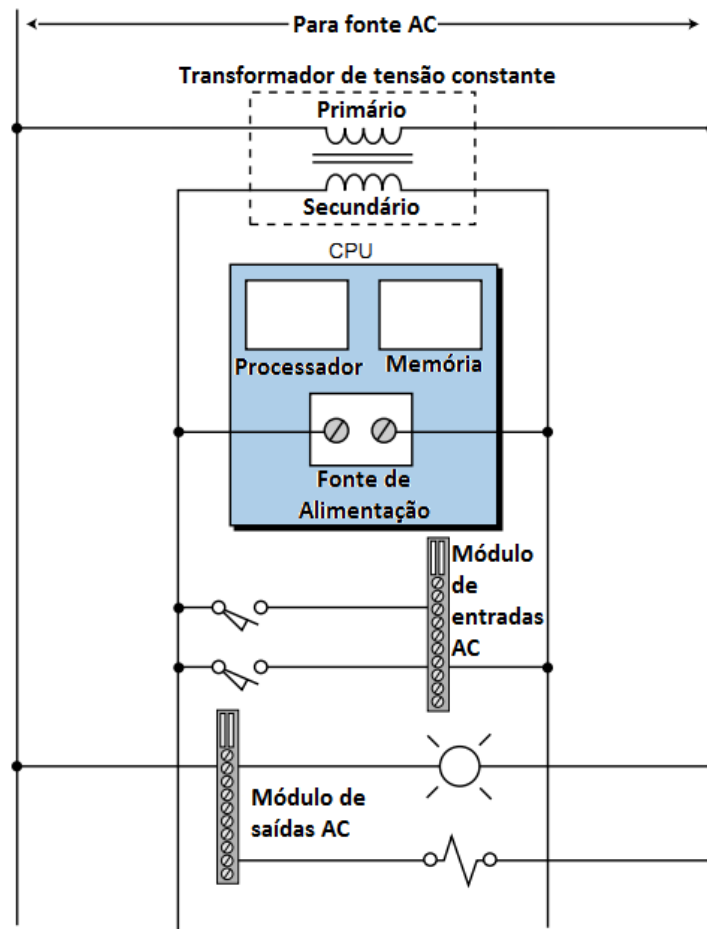


Figura 2.40 - Ligação de um transformador de tensão constante e um PLC [1].

Outro factor que afecta directamente o funcionamento do PLC é a interferência electromagnética (EMI) ou ruído eléctrico. Apesar dos PLC's serem mais robustos que a maior parte dos equipamentos electrónicos, a EMI pode causar vários problemas [9].

Em muitos casos, os PLC's serão instalados em áreas onde a alimentação AC é estável. Contudo, os equipamentos vizinhos podem gerar quantidades consideráveis de interferência electromagnética [1].

Ao colocar o PLC num transformador de isolamento separado dos geradores de emissões EMI, a fiabilidade do sistema aumentará. Um transformador de isolamento não necessita de um transformador de tensão constante; este deve estar localizado entre o PLC e a fonte de alimentação AC [1].

2.1.4.8. Dispositivos de programação

Desde o aparecimento dos PLC's no mercado, os fabricantes têm-se esforçado cada vez mais para conseguir uma *interface* de programação o mais simples possível, para que os utilizadores pudessem passar mais tempo a programar e não a aprender como o fazer.

As linguagens dos PLC's são semelhantes, estando as únicas diferenças associadas a pequenas alterações na utilização de determinadas funções e instruções de cada PLC.

Assim, existem dois tipos de dispositivos de programação [9]:

- Computador Pessoal (PC);
- Miniprogramador ou Terminal Portátil de Programação (HHP).

O PC apresenta-se hoje em dia como uma ferramenta integral do nosso estilo de vida, pois qualquer que seja a tarefa ou necessidade, é rara a vez que tal não implique a utilização de um PC, seja para visualizar o correio electrónico, aceder ao extracto bancário, efectuar compras ou até mesmo comunicar com pessoas e/ou amigos (além da sua utilização para programação de PLC's).

Relativamente aos PLC's, o PC é utilizado como base do programa de *software* para programação dos PLC's. Este *software* permite aos utilizadores criar, editar, armazenar, localizar falhas (*debugging*¹⁷), gerar relatórios e visualização em tempo real do programa [9].

Algumas das vantagens do PC face à utilização de HHP's são as seguintes:

- Maior flexibilidade de utilização (permitem programar PLC's e serem usados para diversas outras tarefas);
- Devido ao maior tamanho dos ecrãs, permitem visualizar de forma mais completa e simples toda a informação e programa do PLC;
- Permitem a ligação a áreas de rede locais, facilitando assim a troca e recolha de informação dos processos que podem ser vitais para implementação de melhorias futuras [1];
- Servem de supervisor de controlo de diversos PLC's ligados entre si numa rede [1] (Fig. 2.41).

Em adição à programação e recolha de dados, os PC's podem ser utilizados para permitir ao programador definir o propósito e função de cada endereço de I/O que é utilizado num PLC. Além disso, os PC's possuem as capacidades adequadas para conseguirem comunicar com os PLC's através de folhas de cálculos e bases de dados pré-definidas,

¹⁷ *debugging* - processo de encontrar e reduzir defeitos ou erros num aplicativo de *software*, permitindo assim uma mais abrangente e eficiente forma de comunicação e recolha de dados entre PC's e PLC's.

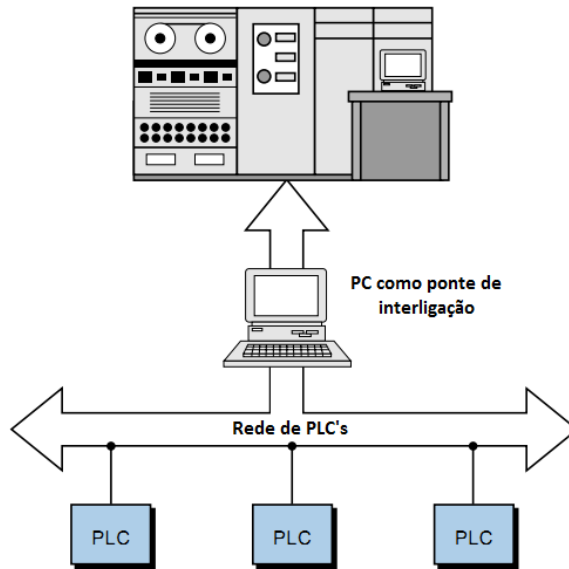


Figura 2.41 - PC como ponte de interligação entre uma *mainframe*¹⁸ e uma rede de PLC's [1].

Relativamente aos HHP's, apesar de poderem ser utilizados para programação de PLC's, são mais frequentemente utilizados na localização de falhas (*debugging*), pois apresentam-se mais compactos e possuem a sua própria memória para armazenamento de programas [9]. São relativamente mais baratos e permitem a programação de pequenos PLC's [até 180 I/O] [1].

Usualmente, este tipo de programadores são construídos para serem compatíveis com dois ou mais produtos de uma determinada família de PLC's e permitem visualizar o funcionamento dos programas e fazer pequenas alterações ao código do mesmo.

Os miniprogramadores podem ser inteligentes ou não-inteligentes. Dispositivos HHP's não-inteligentes podem ser usados para introduzir e editar o programa do PLC com capacidades de monitorização *online* e edição reduzidas. Tais capacidades são reduzidas devido à limitada quantidade de memória e tamanho do ecrã.

Por sua vez, os HHP's inteligentes são baseados em microprocessadores e providenciam aos utilizadores inúmeras vantagens, tais como a capacidade de realizar rotinas de diagnóstico do sistema [1].

¹⁸ *mainframe* - computador de grande porte, dedicado ao processamento de um grande volume de informação.

Alguns HHP's permitem a utilização de cartões de memória para expansão da memória interna do PLC. Este tipo de armazenamento extra é vantajoso quando o programa de controlo de um PLC necessita de ser copiado e posteriormente transferido para outros PLC's [1].

2.1.4.9. Interfaces de operação

Geralmente, o painel frontal de um PLC possui uma série de luzes indicadoras que fornecem indicações referentes à alimentação, operação, falhas e estados das I/O [9]. Para melhorar a interacção entre o utilizador e o PLC surgiram dispositivos de programação (Fig. 2.42) com *displays* gráficos e alfanuméricos, que vieram permitir uma consolidação de todas as funções dos tradicionais dispositivos num único painel de controlo.



Figura 2.42 - Exemplo de um terminal portátil de programação [9].

Estes terminais permitem as seguintes funções:

- Elaboração do programa;
- Análise do conteúdo dos endereços de memória;
- Introdução de novas instruções;
- Modificação do programa existente;
- Monitorização do programa do utilizador;
- Cópia do programa para um disco ou memória externa.

Devido à sua capacidade de apresentar mensagens e exibir texto e informação acerca do estado do programa, estas *interfaces* diminuem a necessidade de treino e formação do programador e reduzem os custos do sistema, componentes e instalação.

2.1.4.10. Linguagens de programação

A linguagem de programação de um PLC é utilizada por engenheiros, técnicos e electricistas e devem portanto, ser utilizadas técnicas e conceitos acessíveis a todos.

Existem vários tipos de linguagem de programação de um PLC, sendo elas:

- Linguagens gráficas:
 - Diagrama *ladder* (LD) ou diagrama de contactos;
 - Gráficos de funções sequenciais (SFC) ou *grafcet*;
 - Diagramas de blocos de funções (FBD).
- Linguagens textuais:
 - Texto estruturado (ST);
 - Lista de instruções (IL).

Cada tipo linguagem possui determinados conjuntos de funções que definem o funcionamento e aplicações de um PLC.

Alguns desses tipos de funções são os seguintes:

- Funções lógicas;
- Memorização;
- Temporização;
- Contagem;
- Movimentação de dados;
- Funções aritméticas;
- Conversão de dados;
- Controladores PID;
- Saltos controlados;
- Comunicação;
- Bases de dados, entre outros.

À medida que a diversidade de PLC's aumenta, aumenta também a complexidade de soluções disponíveis para o mesmo processo, utilizando linguagens próprias e sintaxe de programação sem nenhum critério de padronização [11].

Com o objectivo de estabelecer um padrão para que os *softwares* de programação pudessem processar os comandos, manipular as variáveis e a própria estrutura de apresentação (*interface do software*), foi criado um comité internacional organizado para promover e criar um modelo formal de padronização mundial: a norma IEC 1131-3 [11].

2.1.4.10.1. NORMA IEC 1131-3

A norma IEC 1131-3, desenvolvida juntamente com os fabricantes de PLC's, utilizadores e sistemas académicos, consiste em 5 partes [25]:

- Visão geral;
- Equipamento e requerimentos de teste;
- Linguagens de programação;
- Manuais/guias do utilizador;
- Comunicação.

Esta norma procura juntar as 5 partes acima referidas e tenta criar um modelo genérico de regras a nível mundial com o intuito de promover uma redução de custos relacionados com a programação. O seu objectivo é desenvolver um programa dentro de um ambiente integrado, proporcionando uma maior rigidez de programação, reduzindo a quantidade de erros no desenvolvimento de um programa, aumentando a facilidade de programação e criando documentação e detectando erros [11].

Assim, tornou-se possível padronizar os procedimentos de depuração de erros cometidos durante o desenvolvimento do programa, bem como os erros cometidos durante a alteração do programa.

Um dos maiores benefícios na criação da norma IEC 1131-3 reside na possibilidade de utilização de várias linguagens de programação no mesmo PLC [25]. Isto permite que o programador seleccione a linguagem que mais se adequa ao seu objectivo ou conhecimento.

Outra novidade é o facto de, no próprio ambiente de programação, ser possível trabalhar com o próprio nome quer dos blocos de funções quer de nomes de variáveis de entrada ou saída, memórias auxiliares, *flags*, entre outros e não com um endereço específico dessas instruções, blocos ou funções. Deste modo, a estrutura do programa torna-se mais leve, clara, simples e favorece a leitura do programa e futuras modificações do mesmo [11].

O ambiente de programação pode ser dividido em duas partes: a parte de programação do programa onde se encontram todas as funções, instruções e blocos disponíveis e passíveis de serem utilizados na construção do programa e a parte de declaração de variáveis.

Outra característica da norma IEC 1131-3 é a possibilidade de se definirem variáveis globais e variáveis locais. Assim, um bloco de função pode conter uma variável global na sua estrutura interna, desde que essa variável seja necessária aos restantes blocos do programa. Caso a variável seja utilizada apenas dentro desse bloco específico, apresenta a conotação de variável local [11].

2.1.4.10.2. LINGUAGEM LADDER

A linguagem *ladder* surgiu com o intuito de representar o mais fielmente possível o conjunto de regras, funções e instruções implementadas pelo circuito físico que se pretende controlar.

Na seguinte figura são representados os símbolos utilizados num circuito *ladder* físico e o respectivo circuito equivalente em formato *ladder* de um PLC.

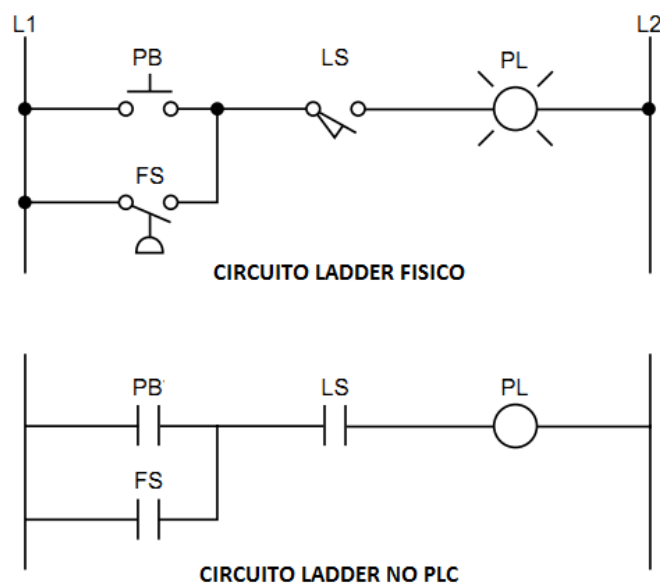


Figura 2.43 - Circuito *ladder* físico vs circuito *ladder* no PLC [1].

A evolução da linguagem *ladder* original tornou a programação *ladder* num poderoso sistema de instruções, pois foram adicionadas novas funções, tais como o controlo básico de relés, temporização e contagem.

Como é possível observar na Fig. 2.43, a linguagem *ladder* utiliza a lógica de relé, com contactos e bobinas, e portanto, pessoas com conhecimento eléctrico facilmente são capazes de perceber o básico deste tipo de linguagem.

Cada uma das linhas horizontais é uma sentença lógica onde os contactos são entradas das sentenças, as bobines as saídas e a interligação dos contactos a lógica [26].

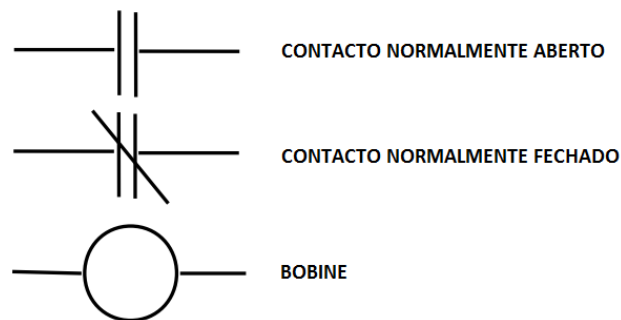


Figura 2.44 - Simbologia básica da linguagem *ladder* [26].

Na linguagem *ladder*, cada operando é identificado com um endereço de memória que é associado ao PLC. Esse endereço aparece no *ladder* com um nome simbólico, de forma a facilitar a programação e a leitura do programa.

No caso particular do autómato utilizado para este projecto, os endereços mais utilizados para programação são os seguintes:

%Ia.b - Endereço referente às entradas digitais do PLC, onde “a” varia consoante o número de *interfaces* ou módulos de entrada ligados ao PLC e “b” consoante o número de entradas disponível em cada módulo ou *interface*. O número de entradas é limitado pela CPU de cada PLC - no caso do S7-1200 1212C permite apenas 1024 *bytes* de endereços de entrada [27].

%Qa.b - Endereço referente às saídas digitais do PLC, onde “a” varia consoante o número de *interfaces* ou módulos de saída ligados ao PLC e “b” consoante o número de saídas disponível em cada módulo ou *interface*. O número de saídas é limitado pela CPU de cada PLC - neste caso particular, permite apenas 1024 *bytes* de endereços de saída [27].

%Ma.b - Endereço referente a *bits* de memória do sistema. “a” varia de 0 até ao número máximo de endereços disponível em cada PLC, no caso do PLC utilizado neste projecto - S7-1200 1212C - 4096 *bytes* de endereços e “b” varia de 0 até 7 (8 *bits*) [27].

%MBa - Referente aos endereços de memória do tipo *byte* (8 *bits*). “a” varia de 0 até 4096 *bytes* no 1212C [27].

%MWa - Referente aos endereços de memória do tipo *word* (16 *bits*). “a” varia de 0 até 4096 *bytes* no 1212C [27].

%MDa - Referente aos endereços de memória do tipo *Double Word* (32 bits). “a” varia de 0 até 4096 bytes no 1212C [27].

%IWa - Referente aos endereços das entradas analógicas. “a” possui apenas dois valores: 64 e 66 (valores de origem). Estes valores podem no entanto ser modificados aquando da criação do programa.

%QWa - Referente aos endereços das saídas analógicas. “a” possui apenas um valor (no caso de se utilizar a placa interna de uma saída analógica): 80 (valor de origem). Este valor pode no entanto ser modificado aquando da criação do programa.

Os tipos de variáveis de dados disponíveis são os seguintes (no caso do PLC S7-1200 da Siemens) [27]:

Tabela 2.5 - Tipos de variáveis de dados: *Bit*, *Byte*, *Word* e *Dword* [27].

Tipo	Tamanho Bit	Tipo Número	Escala Numérica	Exemplos	Exemplo Endereços
<i>Bool</i>	1	<i>Boolean</i>	Falso ou Verdadeiro	Verdadeiro,1	I1.0 Q0.1 DB1.DBX2.3
		<i>Binary</i>	0 ou 1	0, 2#0	
		<i>Octal</i>	8#0 ou 8#1	8#1	
		<i>Hexadecimal</i>	16#0 ou 16#1	16#1	
<i>Byte</i>	8	<i>Binary</i>	2#0 a 2#11111111	2#00001111	IB2 MB10 DB.DBB4
		<i>Unsigned Integer</i>	0 a 255	15	
		<i>Octal</i>	8#0 a 8#377	8#17	
		<i>Hexadecimal</i>	B#16#0 a B#16#FF	B#16#F,16#F	
<i>Word</i>	16	<i>Binary</i>	2#0 a 2#1111111111111111	2#1111000011110000	MW10 DB1.DBW2 Tag_name
		<i>Unsigned Integer</i>	0 a 65535	61680	
		<i>Octal</i>	8#0 a 8#177777	8#170360	
		<i>Hexadecimal</i>	W#16#0 a W#16#FFFF	W#16#F0F0	
<i>DWord</i>	32	<i>Binary</i>	2#0 a 2#11111111111111111111111111111111	2#111100001111111100001111	MD10 DB1.DBD8 Tag_name
		<i>Unsigned Integer</i>	0 a 4294967295	15793935	
		<i>Octal</i>	8#0 a 8#377777777777	8#74177417	
		<i>Hexadecimal</i>	DW#16#0000_0000 a DW#16#FFFF_FFFF	DW#16#F0F0F0F	

Tabela 2.6 - Tipo de variável de dados: *Integer* [27].

Tipo	Tamanho bit	Escala Numérica	Exemplos	Exemplos de endereços
<i>USint</i>	8	0 a 255	78, 2#01001110	MB0, DB1.DBB4, Tag_name
<i>SInt</i>	8	-128 a 127	+50, 16#50	
<i>UInt</i>	16	0 a 65.535	65.295, 0	MW2, DB1.DBW2, Tag_name
<i>Int</i>	16	-32.768 a 32.767	30.000, +30.000	
<i>UDint</i>	32	0 a 4.294,967,295	4.042,322,160	MD6, DB1.DBD8, Tag_name
<i>DInt</i>	32	-2.147,483,648 a 2.147,483,647	-2.131,754,992	

Tabela 2.7 - Tipo de variável de dados: *Floating-point real* [27].

Tipo	Tamanho bit	Escala Numérica	Exemplos	Exemplos Endereços
<i>Real</i>	32	-3.402823e+38 a -1.175495e-38 +1.175495e-38 a +3.402823e+38	123.456, -3.4, 1.0e-5	MD100, DB1.DBD8, Tag_name
<i>LReal</i>	64	-1.7976931348623158e+308 a -2.2250738585072014e-308 +2.2250738585072014e-308 a +1.7976931348623158e+308	12345, 1234569e40, 1.2e840	DB_name.var_name

Tabela 2.8 - Tipos de variáveis de dados: *Time*, *Date*, *Time_of_Day* e DTL [27].

Tipo	Tamanho	Escala	Exemplos
<i>Time</i>	32 bits	T#-24d_20h_31m_23s_648ms a T#24d_20h_31m_23s_647ms Armazenado como: -2.147,483,646ms a +2.147,483,647ms	T#5m_30s T#1d_2h_15m_30s_45ms TIME#10d20h30m20s630ms 500h10000ms
<i>Date</i>	16 bits	D#1990-1-1 a D#2168-12-31	D#2009-12-31 DATE#2009-12-31 2009-12-31
<i>Time_of_Day</i>	32 bits	TOD#0:0:0:1 a TOD#23:59.59.999	TOD#10:20:30.400 TIME_OF_DAY#10:20:30.400 23:10:1
<i>DTL (Date and Time Long)</i>	12 bytes	Min.: DTL#1970-01-01-00:00:00.0 Máx.: DTL#2554-12-31-23:59:59.999 999 999	DTL#2008-12-16-20:30:20.250

Tabela 2.9 - Tipos de variáveis de dados: *Char* e *String* [27].

Tipo	Tamanho	Escala	Exemplos
<i>Character</i>	8 bits	Códigos de caracteres ASCII: 16#00 a 16#FF	'A', 't', '@'
<i>String</i>	n+ 2 bytes	n = (0 a 254 bytes)	'ABC'

Arrays (matrizes) - podem conter no seu interior múltiplos elementos do mesmo tipo. A *syntax*¹⁹ utilizada para a criação de uma matriz através do editor de blocos do *software* STEP 7 V11 deve consistir no seguinte:

"Array [*lo* .. *hi*] of *type*",

onde "*lo*" representa o número inicial mais baixo da matriz, "*hi*" o número mais alto e fim da matriz e "*type*" o tipo de dados que a matriz irá conter [27].

A correcta construção de uma matriz deve perfazer as seguintes regras [27]:

- Todos os elementos da matriz devem ser do mesmo tipo;
- O número inicial da matriz pode ser um número negativo, mas tem que ser sempre menor que o número final da matriz;
- Matrizes podem possuir até 6 dimensões;
- Matrizes multidimensionais devem ser separadas por vírgulas;
- O tamanho de uma matriz = (tamanho de um elemento * número total de elementos da matriz).

Exemplos de declarações de matrizes: ARRAY [1..20] of REAL
 ARRAY [-5..5] of INT
 ARRAY [1..2, 3..4] of CHAR

Exemplos de endereços de matrizes: ARRAY1[0]
 ARRAY2[1,2]

Existem ainda as *structures* (estruturas) que permitem definir estruturas de dados com diversos tipos de dados no seu interior e os *data blocks* (blocos de dados) que possuem a vantagem de, além de permitirem a declaração de variáveis, reter o seu conteúdo mesmo após uma falha de alimentação da CPU, evitando assim perdas indesejadas de dados.

Os *pointers* (ponteiros), como o próprio nome indica, permitem apontar para endereços de memória específicos [27].

A linguagem *ladder* pode ser dividida em 2 tipos de instruções: as instruções básicas e instruções reforçadas/avançadas [1].

Básicas: contacto de relé, saída de relé, temporizador, contador, saltar para/ir para, finalização, adição, subtracção, multiplicação, divisão, comparação e saltos para sub-rotinas.

¹⁹ *syntax* - regras e princípios que governam uma estrutura ou formação de uma palavra.

Avançadas: aritmética de dupla precisão, raiz quadrada, movimentação de registos, movimentação para tabelas de registos, *first in first²⁰ out* (FIFO), *shift²¹* de registo, rotação de registo, bloco de diagnóstico, bloco de transferência (entrada/saída), sequenciador, controlador PID, comunicação/rede e lógica de matrizes.

O princípio básico da linguagem *ladder* é controlar saídas através de operações baseadas em condições de entrada.

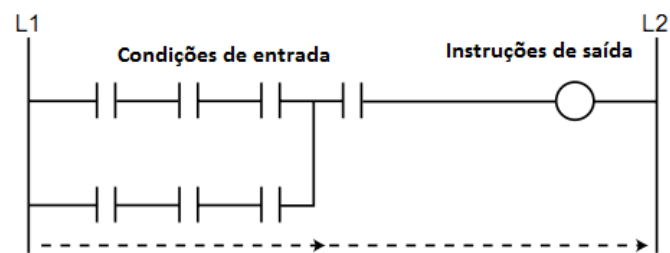


Figura 2.45 - Estrutura de uma *rung*²² [1].

Caso as condições de entrada sejam todas satisfeitas (1 - Verdadeira), a alimentação circula até à saída e activa-a (Fig. 2.46). Caso contrário, a saída não é activada.

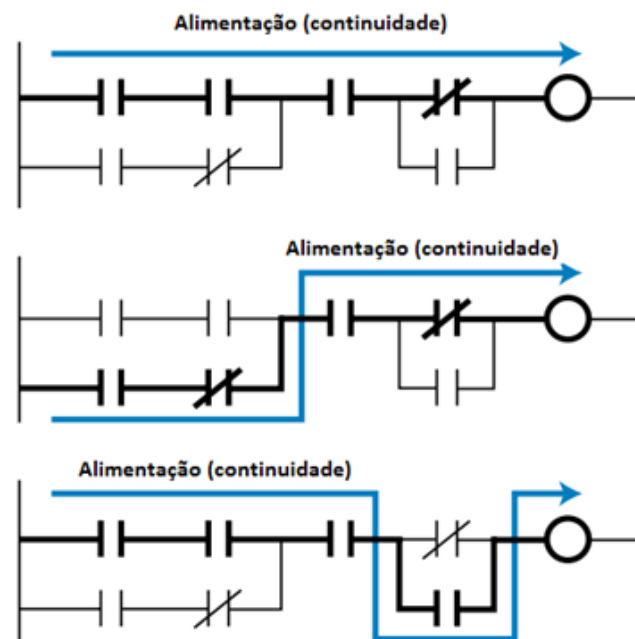


Figura 2.46 - Caminhos de continuidade numa *rung* [1].

²⁰ *first in first out* - refere-se a estruturas de dados; geralmente utilizado para implementar filas de espera.

²¹ *shift* - operador aritmético; mudar ou deslocar um *bit* ou registo.

²² *rung* - linha de programação num diagrama de *ladder* lógico. Cada *rung* controla uma saída.

O formato das *rungs* e contactos nelas existentes depende do objectivo do programa. Os contactos podem ser colocados em série (todos seguidos na horizontal) ou em paralelo (paralelos uns aos outros).

Segue-se uma lista com as funções básicas e algumas das mais utilizadas funções avançadas existentes no *software* TIA Portal STEP7 V11 SP2 utilizado para a realização deste projecto.

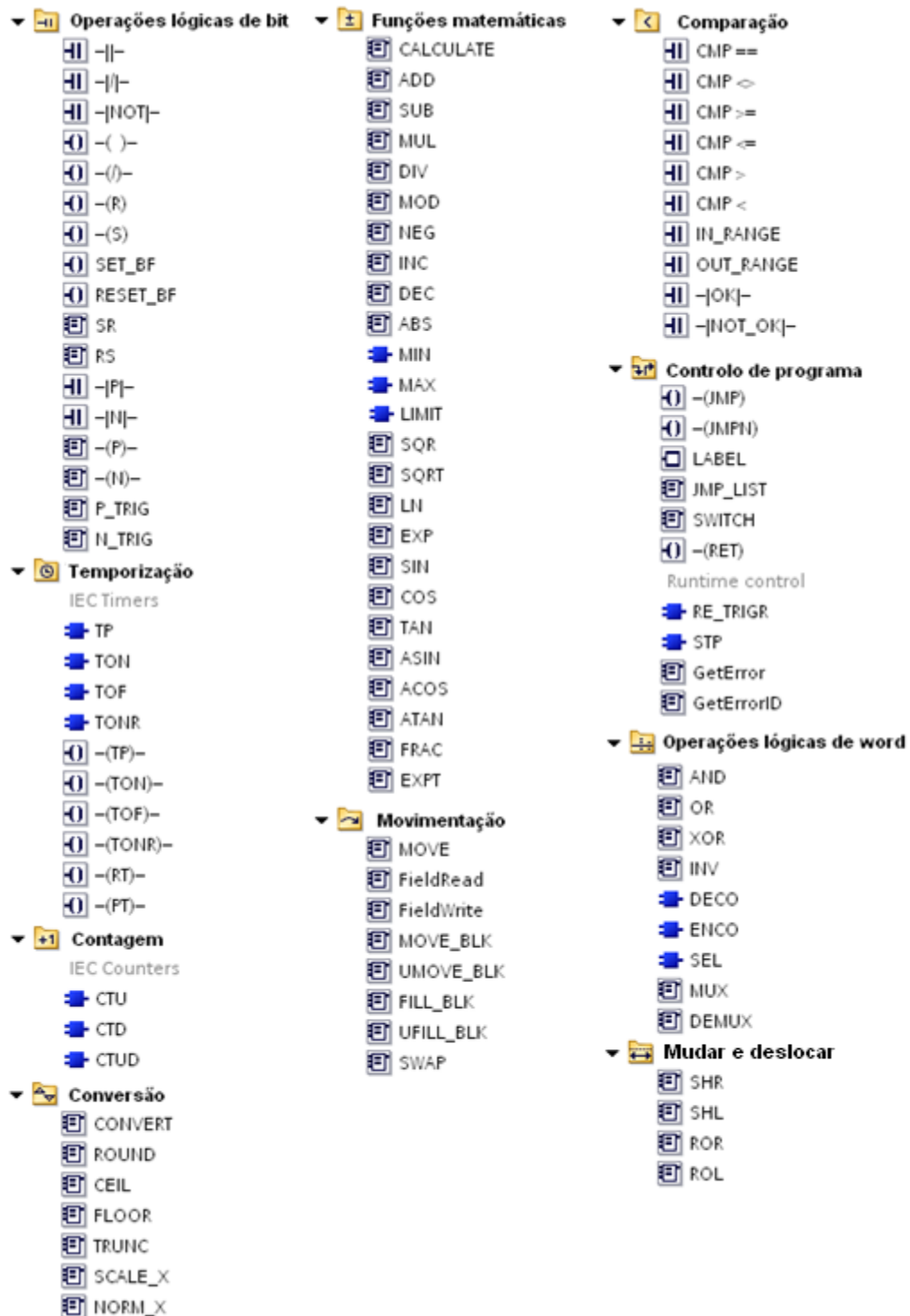


Figura 2.47 - Lista das funções mais utilizadas no *software* STEP 7 V11.

2.1.4.10.3. LINGUAGEM BOOLEANA

Em 1849, George Boole desenvolveu a álgebra de Boole. O objectivo desta álgebra era auxiliar a lógica do raciocínio, pois trouxe uma maior facilidade e simplicidade na escrita de combinações de instruções lógicas complicadas [1].

Alguns fabricantes de PLC's utilizam linguagem booleana para controlar os autómatos. A linguagem booleana utiliza álgebra booleana (explicada mais à frente) para introduzir e explicar a lógica de controlo, ou seja, utiliza funções *AND*, *OR* e *NOT* para implementar os circuitos de controlo no programa [1].

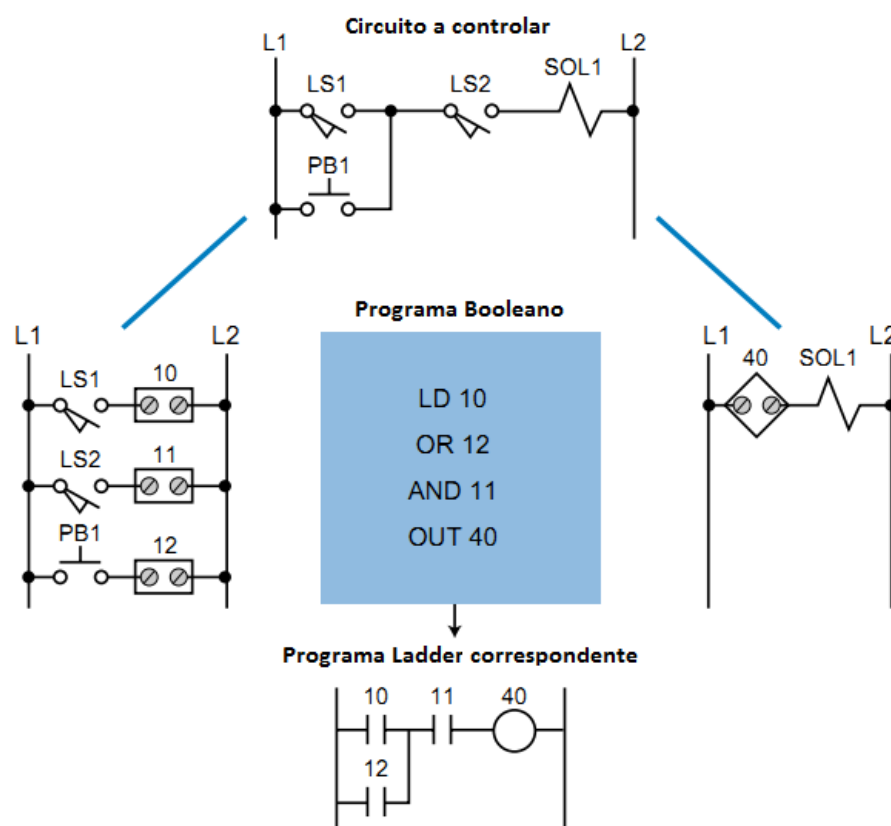


Figura 2.48 - Programa em linguagem booleana e correspondente em *ladder* [1].

A figura seguinte representa os operadores básicos da lógica booleana, que estão relacionados com as funções básicas *AND*, *OR* e *NOT* da lógica digital.




Simbolo Lógico	Afirmção Lógica	Equação Booleana
	Y é 1 se A AND B forem 1	$Y = A \cdot B$ or $Y = AB$
	Y é 1 se A OR B for 1	$Y = A + B$
	Y é 1 se A for 0 Y é 0 se A for 1	$Y = \bar{A}$

Figura 2.49 - Relação da álgebra booleana com as funções AND, OR e NOT [1].

As portas básicas que implementam as funções lógicas simples são as seguintes:

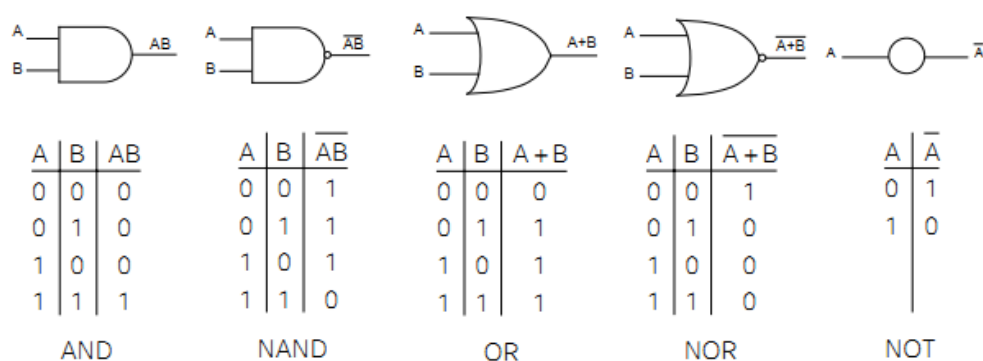


Figura 2.50 - Portas básicas da lógica booleana [1].

As portas básicas podem ser combinadas para implementar funções lógicas mais complexas.

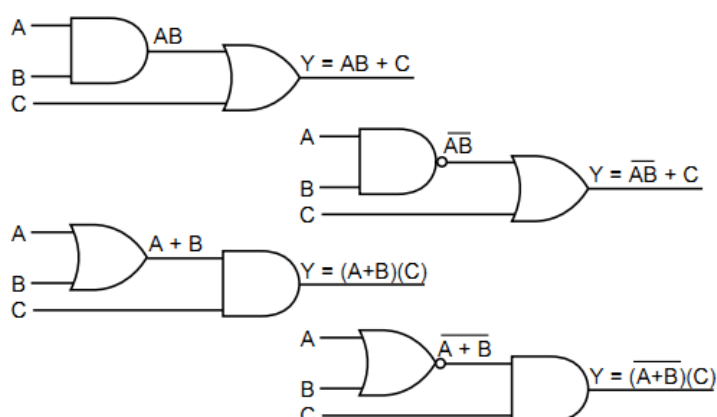


Figura 2.51 - Combinação das portas lógicas básicas booleanas [1].

Este tipo de lógica responde a uma série de leis e regras designadas de regras da álgebra de Boole.

Leis Comutativas

$$A + B = B + A$$

$$AB = BA$$

Leis Associativas

$$A + (B + C) = (A + B) + C$$

$$A(BC) = (AB)C$$

Leis de Morgan

$$\overline{(A + B)} = \bar{A}\bar{B}$$

$$\overline{(\bar{A}\bar{B})} = A + B$$

$$\bar{\bar{A}} = A, \bar{1} = 0, \bar{0} = 1$$

$$A + \bar{A}B = A + B$$

$$AB + AC + B\bar{C} = AC + B\bar{C}$$

Leis Distributivas

$$A(B + C) = AB + AC$$

$$A + BC = (A + B)(A + C)$$

Lei da Absorção

$$A(A + B) = A + AB = A$$

Figura 2.52 - Operações lógicas utilizando álgebra de Boole [1].

Na tabela seguinte, estão apresentadas as típicas funções utilizadas aquando da programação em linguagem booleana.

Tabela 2.10 - Instruções booleanas e respectivas instruções em *ladder* [1].

Mnemónica Booleana	Função	Equivalente Ladder
LD/STR	Inicia sequência com contacto NO	
LR/STR NOT	Inicia sequência com contacto NC	
AND	Ligação de contactos NO em série	
AND NOT	Ligação de contactos NC em série	
OR	Ligação de contactos NO em paralelo	
OR NOT	Ligação de contactos NC em paralelo	
OUT	Termina sequência com bobine	
OUT NOT	Termina sequência com bobine negada	
OUT CR	Termina sequência com saída interna	
OUT L	Termina sequência com saída bloqueada	
OUT U	Termina sequência com saída desbloqueada	
TMR	Termina sequência com temporizador	
CTU	Termina sequência com temporizador UP	
ADD	Termina sequência com a função adição	
SUB	Termina sequência com a função subtração	
MUL	Termina sequência com a função multiplicação	
DIV	Termina sequência com a função divisão	
CMP	Termina sequência com a função comparação	
JMP	Termina sequência com a função saltar	
MCR	Termina sequência com saída MCR	
END	Termina sequência com a função terminar	
ENT	Utilizada para introdução de valores nos registos	

2.1.4.10.4. LINGUAGEM GRAFCET

A linguagem *grafcet* é uma linguagem gráfica simbólica que surgiu em França e que representa o controlo de um programa através de passos ou estados no processo.

Esta linguagem providencia uma sequência de eventos que ocorrem em cada estado do programa. Para tal, são utilizados três componentes: passos, transições e acções [1].

A programação *grafcet* tem algumas regras de evolução que passam pela inicialização, transposição de uma transição, evolução das etapas activas, simultaneidade na transposição das transições e prioridade de activação [10].

Regra 1 - Inicialização: na inicialização do sistema activam-se todas as etapas iniciais. Estas etapas dão início ao ciclo de funcionamento do automatismo.

Regra 2 - Transposição de uma transição: uma transição pode ser válida ou não. É válida quando as imediatamente anteriores à transição estão activas. Quando a transição é válida, e a respectiva receptividade verdadeira, a transição é obrigatoriamente transposta.

Regra 3 - Evolução das etapas activas: a transposição de uma transição implica a activação das etapas que estão imediatamente a seguir à transição e à desactivação simultânea das etapas que estavam activas imediatamente antes da transição.

Regra 4 - Simultaneidade na transposição das transições: várias transições podem ser transpostas simultaneamente quando todas as condições para tal se verifiquem.

Regra 5 - Prioridade de activação: caso, no curso do funcionamento, uma etapa seja activada e desactivada ao mesmo tempo, a prioridade é dada à activação.

Poucos PLC's permitem ser programados directamente em *grafcet*. Contudo, vários fabricantes de *software* em linguagem *grafcet* apresentam a possibilidade de programação *offline* em *grafcet*, e posteriormente utilizar um conversor para converter o programa em *grafcet* para linguagem booleana ou *ladder* [1].

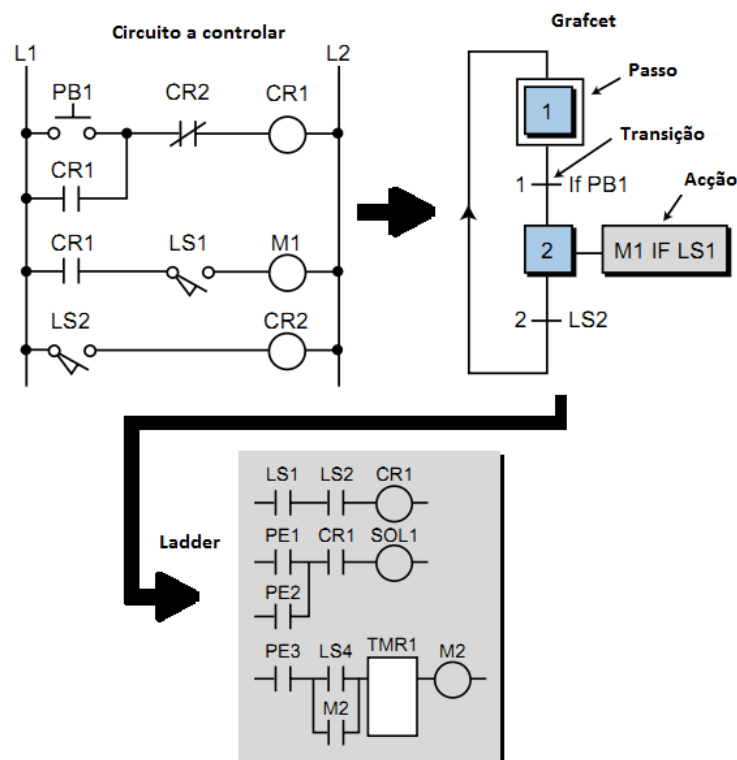


Figura 2.53 - Programa em *grafcet* e conversão para *ladder* (NOTA: o circuito *ladder* convertido é meramente exemplificativo e não corresponde ao circuito *grafcet* representado) [1].

Apesar das linguagens *grafcet* e SFC poderem ser consideradas iguais, pois operam da mesma forma e possuem o mesmo tipo de linguagem, apresentam contudo uma grande diferença: a linguagem *grafcet* utiliza apenas expressões escritas, tais como ABRIR_VARIAVEL para implementar os seus blocos de acção e ligar/desligar dispositivos.

Por outro lado, o SFC implementa acções recorrendo a um maior número de alternativas através da utilização de linguagem *ladder*, lista de instruções, texto estruturado ou diagrama de blocos de funções, permitindo ainda combinar todas estas linguagens, incluindo funções de blocos criadas pelo utilizador [1].

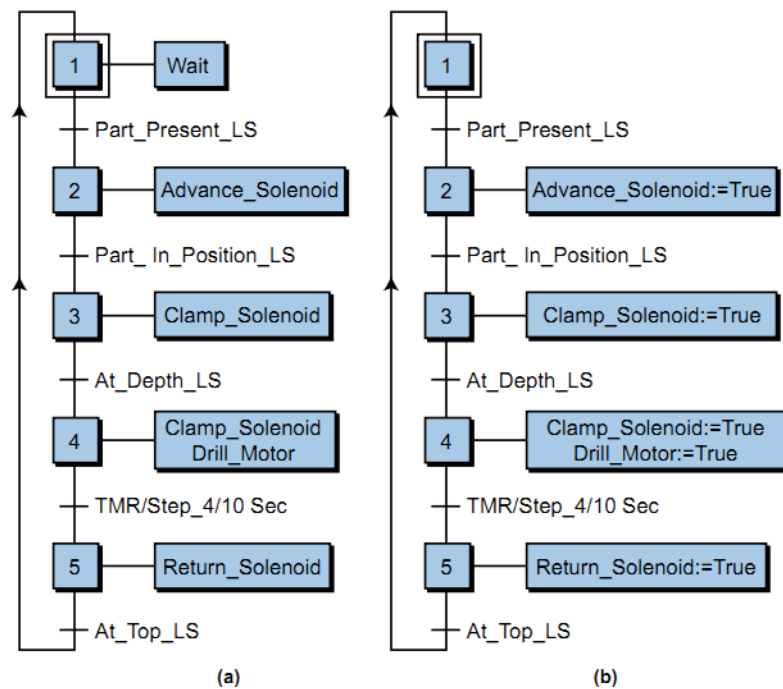


Figura 2.54 - Comparação entre linguagem *grafcet* (a) e linguagem SFC (b) [1].

Uma grande vantagem em utilizar este tipo de linguagem reside na fácil detecção de eventuais problemas que possam surgir (*debugging*). Tendo em conta a figura anterior, facilmente se detecta qual o problema se o programa parar e não passar, por exemplo para o passo 4. O problema, neste caso, seria que a condição *At_Depth_LS* não se verificou. Além disso, a própria linguagem indica o ponto onde o erro ocorreu [1].

2.1.4.10.5. DIAGRAMAS DE BLOCOS DE FUNÇÕES

Este tipo de linguagem (Fig. 2.55) é considerada gráfica e permite ao utilizador programar e ligar elementos de uma forma semelhante aos elementos constituintes de um circuito eléctrico.

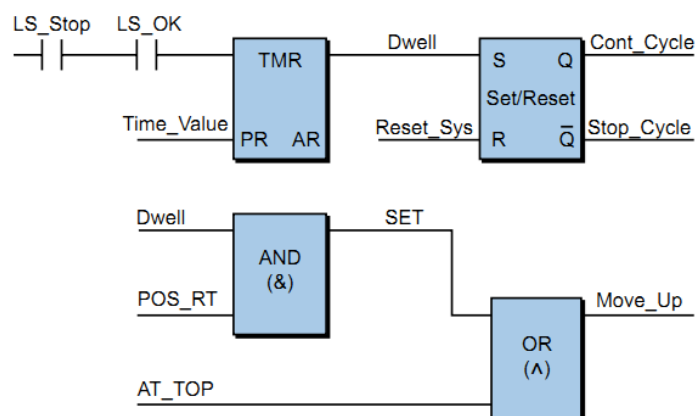


Figura 2.55 - Exemplo de programa em FBD [1].

Adicionalmente, e com a introdução da norma IEC 1131-3, os utilizadores passaram a ter a possibilidade de criar os seus próprios blocos de funções de acordo com as suas necessidades [1]. Tal facto, permitiu uma maior flexibilidade e eficiência na criação de programas.

O aparecimento da norma IEC 1131-3 possibilitou também a capacidade de “misturar” vários tipos de linguagem, ou seja, o utilizador pode criar os seus próprios blocos de funções recorrendo a diferentes tipos de linguagem que não o FBD [1].

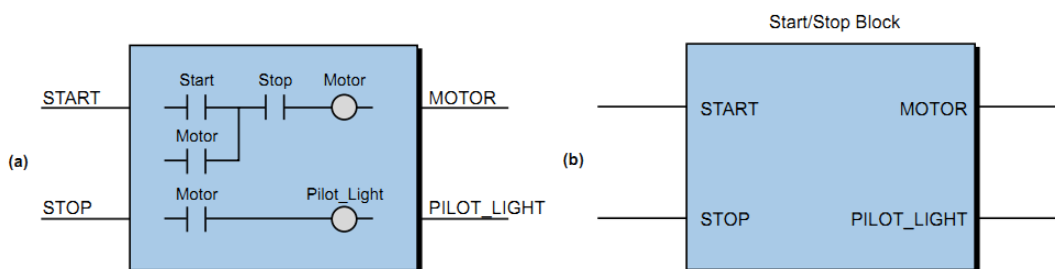


Figura 2.56 - (a) Bloco de função programado em *ladder* (b) Bloco Start/Stop em FBD [1].

2.1.4.10.6. LISTA DE INSTRUÇÕES

A linguagem baseada na lista de instruções é considerada de baixo nível, similar à linguagem de máquina ou linguagem *assembly*²³ utilizada nos microprocessadores. Este tipo de linguagem é bastante útil quando é necessário criar pequenos programas cuja optimização e rapidez devam ser elevadas.

É baseada nas regras da álgebra de Boole; consiste num conjunto de instruções representadas em mnemónicas que indicam as acções ou operações que o programa executa, tais como AND, OR, funções de comparação, funções de temporizadores ou contadores, etc. [10].

O programa em lista de instruções é constituído por conjuntos de linhas, com uma determinada ordem, escritas com as instruções do autómato que se vai utilizar e inicia-se com a instrução *Load* ou *Block* que é introduzida na memória do PLC linha a linha [10].

Um exemplo deste tipo de linguagem é apresentado de seguida.

²³ *assembly* - código de máquina; linguagem de baixo nível. Geralmente utilizado na programação de microprocessadores.

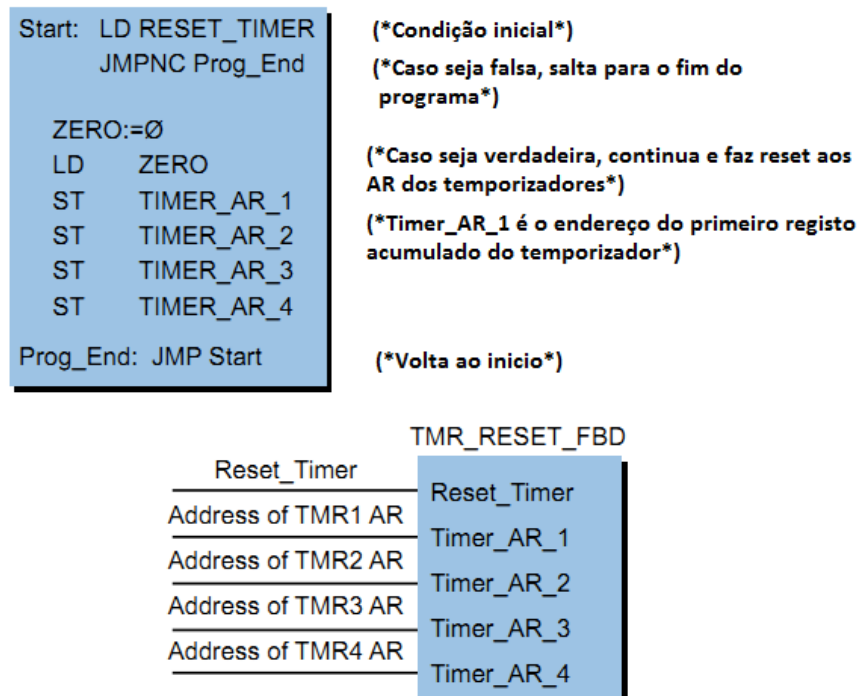


Figura 2.57 - Bloco de função em linguagem de lista de instruções [1].

2.1.4.10.7. TEXTO ESTRUTURADO

Ao contrário do tipo de linguagem anterior, o texto estruturado representa uma linguagem de alto nível que permite uma programação estruturada, isto é, várias tarefas complexas podem ser “partidas” em tarefas mais simples. Este tipo de linguagem recorre a linguagem de computador (PASCAL e BASIC) que utiliza sub-rotinas para executar diferentes partes do código do programa e passar parâmetros e valores entre as diferentes secções do programa. Suporta iterações, tais como *WHILE...DO* e *REPEAT...UNTIL*, assim como outras execuções condicionais tais como *IF...THEN...ELSE* [1]. Além disso, suporta também operações booleanas (*AND*, *OR*, *NOT*, etc.).

```

IF Manual AND NOT Alarm THEN
    Level:=Manual_Level;
    Mixer:=Start AND NOT Reset
ELSE_IF Other_Mode THEN
    Level:=Max_Level;
ELSE    Level:=(Level_Indic × 100)/Scale;
END_IF;

```

Figura 2.58 - Exemplo de linguagem BASIC [1].

Assim como os tipos de linguagem anteriores, esta permite a criação de blocos de funções que se adaptam às necessidades de cada programador.

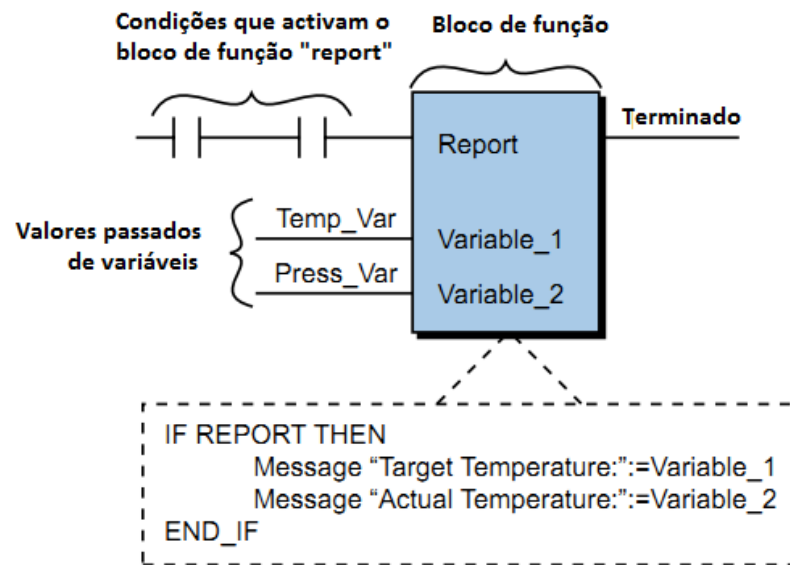


Figura 2.59 - Bloco de função criado utilizando linguagem em texto estruturado [1].

No capítulo seguinte será apresentado o conceito geral do sistema desenvolvido para o projecto de sistema de rega inteligente.

Será também efectuada uma introdução aos componentes utilizados e uma explicação de todos os detalhes e modos de funcionamento do sistema.

O capítulo termina com potenciais melhorias a ser implementadas ao sistema no futuro.

3. Capítulo 3

3.1. Conceito do sistema

O sistema desenvolvido e apresentado nesta dissertação refere-se a um sistema de rega inteligente. Este sistema permite o controlo e monitorização da quantidade de água num reservatório e a regulação do valor do pH. A água é proveniente de um furo, poço ou outro qualquer ponto de captação de água. A sua distribuição, neste caso específico, é realizada por gravidade (tanque situado a uma altura superior relativamente aos circuitos de rega).

Posteriormente, a água é distribuída pelos vários circuitos de rega. Além da monitorização e controlo local do sistema através da consola HMI, foi também elaborada uma página *web* que permite o mesmo tipo de controlo e monitorização mas remotamente (via *internet*).

Adicionalmente, o sistema foi concebido para possibilitar a integração de um sistema de monitorização e controlo de determinadas variáveis referentes ao solo, tais como temperatura, humidade e acidez. Este sistema, referido com maior detalhe na secção 3.1.6.2 desta dissertação, pode ser implementado com recurso à comunicação *ZigBee* e *GPRS*.

O sistema foi criado com o intuito de ser aplicado a um sistema real que surgiu no âmbito de um projecto requisitado por um cliente da empresa TecnoSoares, Lda onde me encontro a estagiar à data de realização desta dissertação. Assim, foi aproveitada a oportunidade de elaborar o projecto para o cliente e realizar a actual dissertação de Mestrado com base na criação desse sistema.

A implementação do sistema de rega inteligente implementado nesta dissertação possui bastantes vantagens face a um controlo manual:

- Controlo autónomo do nível da água e valor do pH;
- Controlo e monitorização geral de todo o sistema no local (consola HMI);
- Possibilidade de controlar e monitorizar todo o sistema à distância (via página *web*);
- Maior facilidade na detecção de avarias dos diversos componentes constituintes do sistema (bomba de água, bomba do pH, etc.);
- Sistema de alarme visível aquando de alguma falha no funcionamento do sistema;
- Melhor controlo dos gastos e maior poupança energética;
- Possibilidade de criar bases de dados com informação recolhida periodicamente, para eventuais comparações estatísticas;
- Sistema preparado para eventuais melhorias futuras e possibilidade de adição de módulos adicionais.

O sistema possui diversos sensores que recolhem informações referentes ao nível da água do reservatório, valor do pH, entre outros, que posteriormente são enviadas e analisadas pelo PLC. Possui também diversos modos de funcionamento (explicados com maior detalhe na secção seguinte) que permitem um melhor e mais eficiente controlo dos gastos de água.

3.1.1. Características gerais

O sistema é composto por diversos componentes, sendo eles:

- Uma bomba de captação de água:
 - ✓ Um sensor que detecta a presença de água na bomba;
- Um reservatório de água com um volume total útil de 3 m³:
 - ✓ Três sensores de nível: nível máximo, mínimo e constante;
 - ✓ Uma electroválvula que controla a saída de água para os circuitos de rega;
 - ✓ Um caudalímetro que dá um impulso eléctrico a cada 100 litros de água utilizados;
 - ✓ Cinco electroválvulas que controlam a abertura ou fecho dos diversos circuitos de rega;
- Um reservatório para armazenamento do reagente regulador do pH:
 - ✓ Uma bomba que introduz o reagente regulador do pH no tanque quando necessário;
 - ✓ Um sensor de nível que detecta o nível mínimo do reagente no reservatório.
 - ✓ Um conversor de frequência (ou variador electrónico) ligado a uma saída analógica do PLC que controla a velocidade da bomba doseadora conforme o maior ou menor desvio do nível de pH da água com o nível definido pelo utilizador.

Relativamente ao funcionamento do sistema, este apresenta 4 modos distintos de funcionamento:

Modo 1 (Manual) - Permite o controlo manual através da consola ou página *web* dos equipamentos constituintes do sistema. Este modo permite o controlo manual da velocidade da bomba doseadora de pH, pelo que o controlo automático do pH é automaticamente desactivado quando este modo de funcionamento é activado.

Modo 2 (Automático por nível) - Controlo automático do nível de água presente no tanque. O utilizador apenas necessita de definir o set-point do nível de água que pretende. Quando o nível de água atingir o set-point definido, a bomba de água é ligada e só pára quando o tanque estiver cheio.

Modo 3 (Automático por diferencial) - Semelhante ao modo anterior, mas com a vantagem de permitir a definição de um valor de diferencial. O utilizador define um set-point (o mesmo que no modo 2) e um valor de diferencial. A bomba de água liga quando se atingir o valor set-point - diferencial ($SP - D$) e pára quando set-point + diferencial ($SP + D$) é atingido.

Modo 4 (Modo nocturno) - Este modo foi desenvolvido com o intuito de poupança energética. A bomba de água só liga durante o período nocturno (período em que a electricidade é mais barata - contador bi-horário) de funcionamento (das 24h às 6h). Se por algum motivo o tanque atingir o nível mínimo de água durante o dia (período em que a bomba de água não funciona), o sistema passa automaticamente para o modo 2 de funcionamento, evitando assim que o tanque fique sem água em qualquer situação.

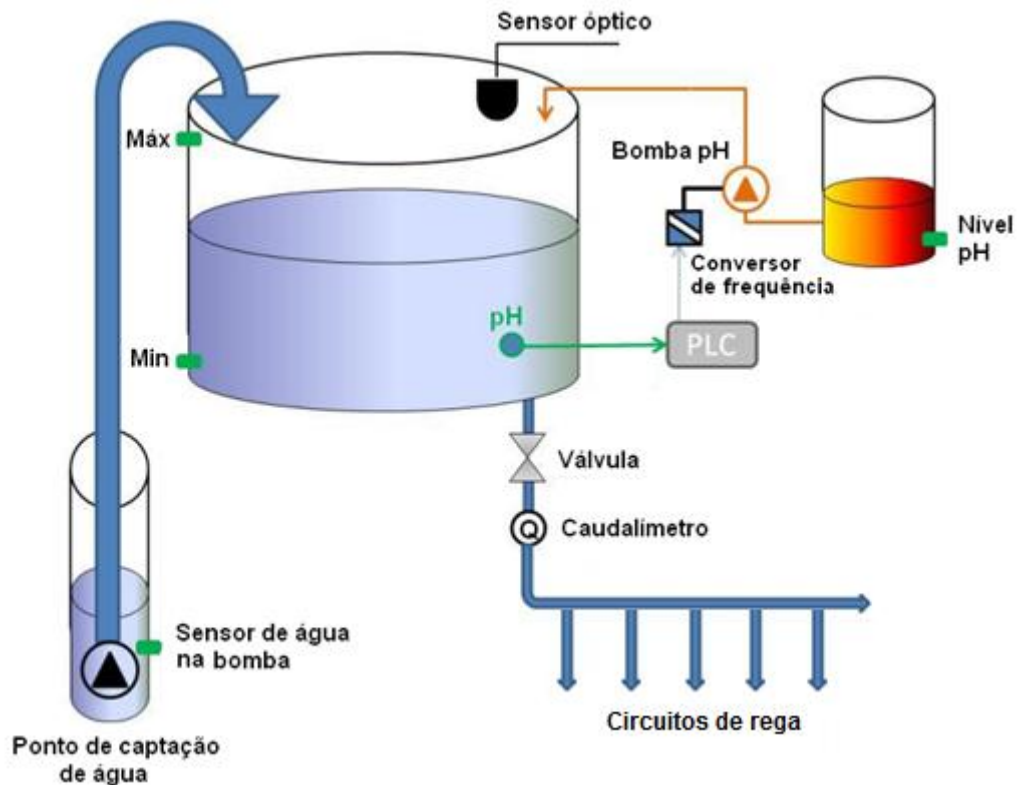


Figura 3.1 - Esquema do sistema desenvolvido.

3.1.2. Introdução ao Siemens S7-1200 1212C

Para o desenvolvimento deste sistema, foi escolhido um autômato da Siemens, mais especificamente o modelo S7-1200 1212C DC/DC/DC.

A escolha deste PLC derivou essencialmente dos seguintes aspectos:

- Número de entradas e saídas (analógicas e digitais) ideais para a implementação do sistema em causa;
- Elevada compatibilidade com diversos componentes, tais como consolas HMI;
- Elevada capacidade de expansão e conectividade, permitindo assim a constante evolução e melhoria do sistema;
- Possibilidade de criação de página na *web* sem necessidade de módulos adicionais;
- Elevada robustez, flexibilidade e preço economicamente acessível.

Este PLC combina um microprocessador, circuitos de entrada e saída, comunicação PROFINET (comunicação TCP/IP - *ethernet*) incorporada, I/O de controlo de alta velocidade e entradas analógicas incorporadas.

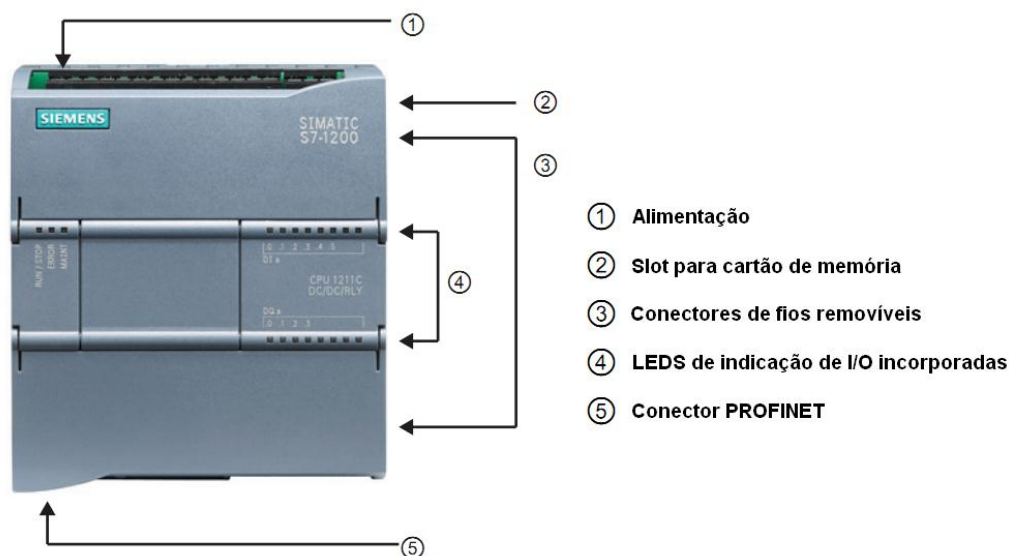


Figura 3.2 - Constituição básica do S7-1200 [27].

Além da comunicação PROFINET, permite ainda, com a adição do respectivo módulo, comunicação através de PROFIBUS, GPRS, RS485 ou RS232 [27].

Este autômato permite também o acesso ao seu conteúdo através de uma *interface web*.

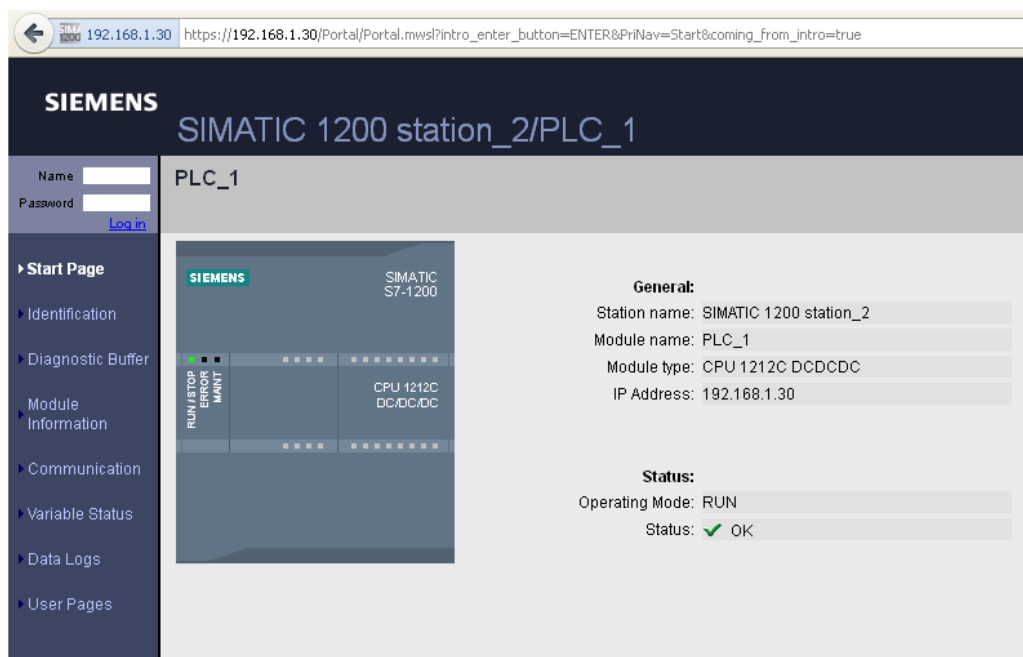


Figura 3.3 - Interface Web Server.

Esta funcionalidade permite efectuar o *login* do utilizador, aceder às informações essenciais do PLC, estado de funcionamento do PLC, utilização de memória, informações de módulos, comunicação, estado das variáveis, relatórios de dados (*data logs*) armazenados na memória interna e ainda a uma página *web* personalizada.

Permite também, e como referido anteriormente, a criação de *data logs* em formato .csv (Excel) para armazenamento periódico de informação e posterior comparação de dados e valores.

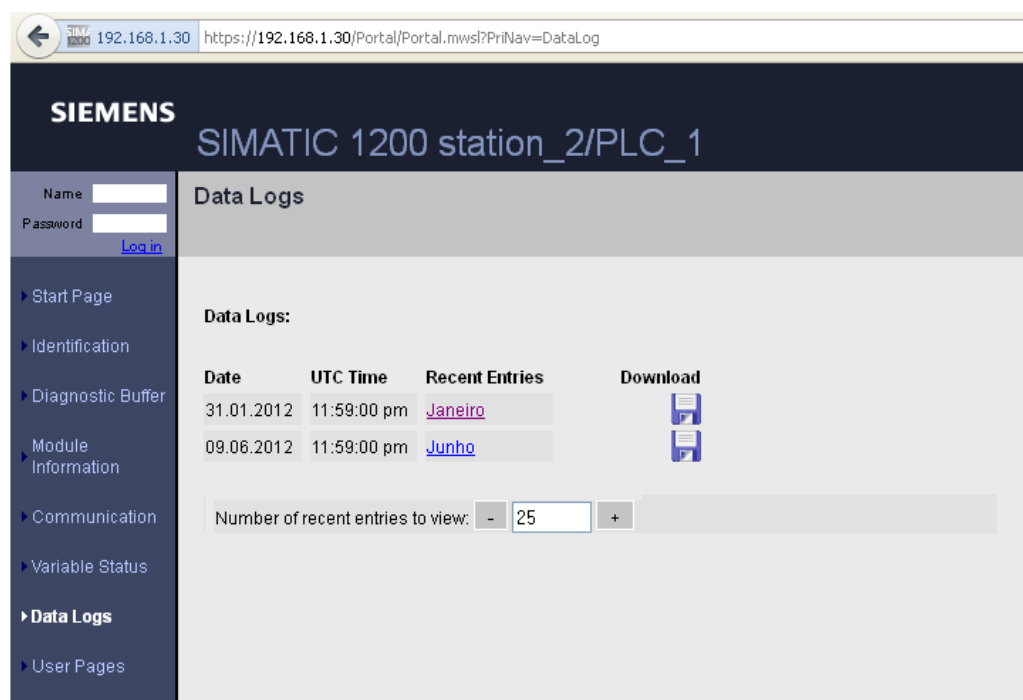


Figura 3.4 - Visualização e *download* dos *data logs*.

Para se conseguir apagar os *data logs* da memória interna do PLC, é necessário *login* de administrador.

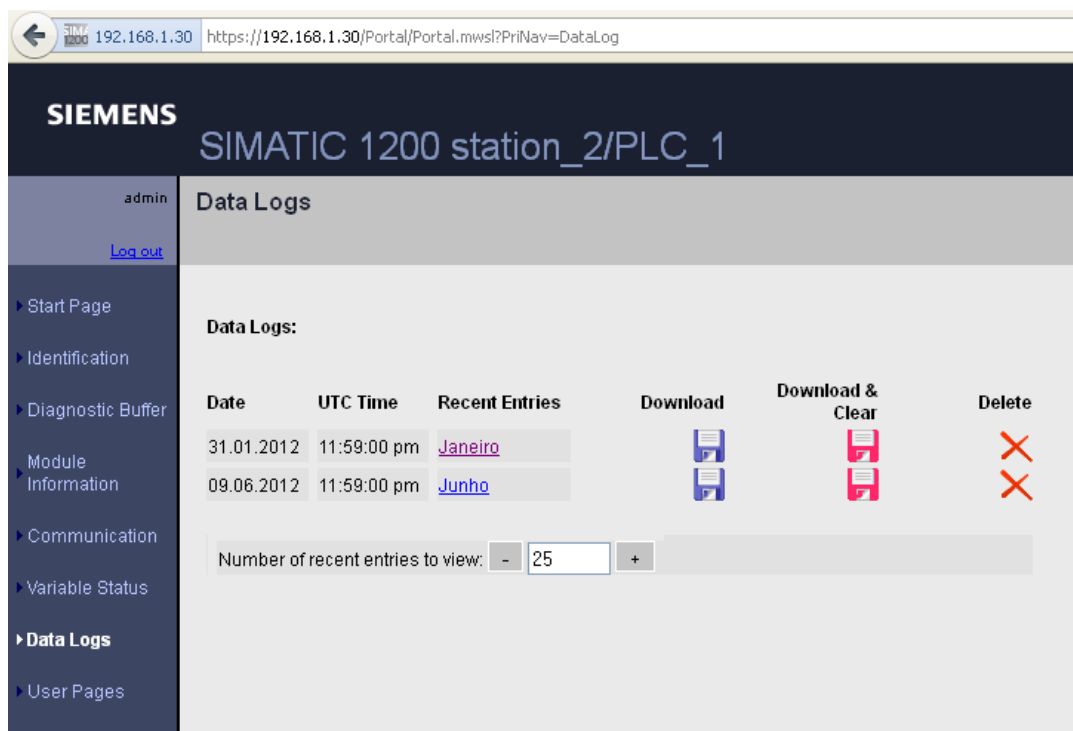


Figura 3.5 - Visualização, *download* e eliminação dos *data logs* com *login* de administrador.

Para outras funções que requeiram alteração do conteúdo interno do PLC tais como forçar valores de variáveis, alteração de conteúdos ou valores, etc., é também necessário efectuar *login* como administrador.

3.1.2.1. Módulos e especificações

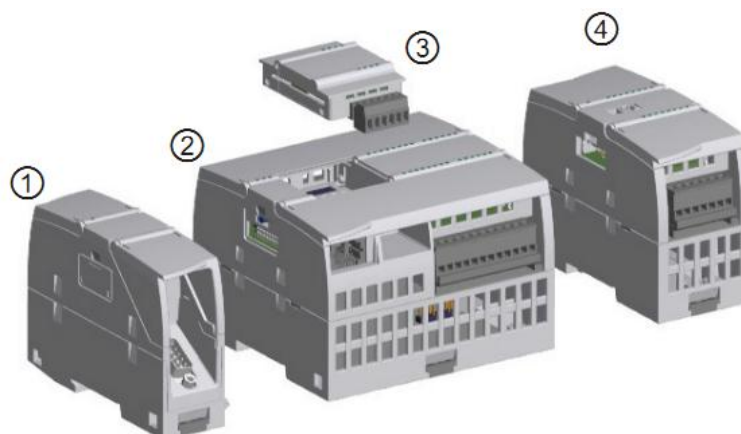
Para além das já referidas características, esta CPU permite a protecção ao seu acesso interno, assim como a protecção através de *password* ao acesso do programa armazenado no PLC, evitando assim a possibilidade de cópias não autorizadas e reprodução/alteração de *software* por parte de terceiros.

Como se pode observar pelo modelo, este apresenta a CPU 1212C da gama com as seguintes características [27]:

Tabela 3.1 - Características da CPU 1212C [27].

Dimensões físicas	90 x 100 x 75 (mm)	
Memória	Trabalho	25 Kbytes
	Carga	1 Mbyte
	Retentiva	2 Kbytes
I/O incorporadas	Digital	8 entradas/6 saídas
	Analógica	2 entradas
Tamanho de processamento de imagem	Entradas (I)	1024 bytes
	Saídas (Q)	1024 bytes
Bit de memória (M)	4096 bytes	
Módulos de expansão de sinal (SM)	2	
Placa de sinal incorporada (SB) ou placa de comunicação (CB)	1	
Módulos de comunicação (CM)	3	
Contadores de alta velocidade	Total	4
	Fase única	3 a 100 kHz 1 a 30 kHz
	Quadratura de fase	3 a 80 kHz 1 a 20 kHz
Saída de pulso (PWM)	2	
Cartão de memória	Cartão SIMATIC (opcional)	
Retenção do relógio interno em tempo real	10 dias, tipicamente / 6 dias no mínimo a 40° C	
PROFINET	1 porta de comunicação <i>ETHERNET</i>	
Velocidade de execução de instruções aritméticas	18 µs / instrução	
Velocidade de execução de instruções booleanas	0.1 µs / instrução	

Permite também, e como referido anteriormente, o acoplamento e ligação de vários módulos adicionais de entradas e saídas e de comunicação.



- ① Módulo de comunicação (CM), processador de comunicação (CP) ou adaptador TS (Teleservice)
 ② CPU
 ③ Placa de sinal incorporada (SB) ou placa de comunicação incorporada (CB)
 ④ Módulo de sinal (SM)

Figura 3.6 - Tipos de módulos de expansão permitidos [27].

Tabela 3.2 - Módulos de entradas e saídas digitais [27].

Tipo	Entradas (IN)	Saídas (OUT)	Entradas/Saídas
③ SB digital	<ul style="list-style-type: none"> 4 × 24 VDC IN, 200 kHz 4 × 5 VDC IN, 200 kHz 	<ul style="list-style-type: none"> 4 × 24 VDC OUT, 200 kHz 4 × 5 VDC OUT, 200 kHz 	<ul style="list-style-type: none"> 2 × 24 VDC IN / 2 × 24 VDC OUT 2 × 24 VDC IN / 2 × 24 VDC OUT, 200 kHz 2 × 5 VDC IN / 2 × 5 VDC OUT, 200 kHz
④ SM digital	<ul style="list-style-type: none"> 8 × 24 VDC IN 	<ul style="list-style-type: none"> 8 × 24 VDC OUT 8 × saídas a relé 8 × saídas a relé 	<ul style="list-style-type: none"> 8 × 24 VDC IN / 8 × 24 VDC OUT 8 × 24 VDC IN / 8 × saídas a relé 8 × 120/230 VAC IN / 8 × saídas a relé
	<ul style="list-style-type: none"> 16 × 24 VDC IN 	<ul style="list-style-type: none"> 16 × 24 VDC OUT 16 × saídas a relé 	<ul style="list-style-type: none"> 16 × 24 VDC IN / 16 × 24 VDC OUT 16 × 24 VDC IN / 16 × saídas a relé

Tabela 3.3 - Módulos de entradas e saídas analógicas [27].

Tipo	Entradas (IN)	Saídas (OUT)	Entradas/Saídas
③ SB analógico	<ul style="list-style-type: none"> 1 × 12 entradas analógicas 	<ul style="list-style-type: none"> 1 × saídas analógicas 	-
④ SM analógico	<ul style="list-style-type: none"> 4 × entradas analógicas 8 × entradas analógicas <p><i>Thermocouple:</i></p> <ul style="list-style-type: none"> - 4 × 16 bits TC - 8 × 16 bits TC <p><i>RTD:</i></p> <ul style="list-style-type: none"> - 4 × 16 bits RTD - 8 × 16 bits RTD 	<ul style="list-style-type: none"> 2 × saídas analógicas 4 × saídas analógicas 	<ul style="list-style-type: none"> 4 × entradas analógicas / 2 × saídas analógicas

Tabela 3.4 - Módulos de comunicação [27].

	Tipo	Descrição
① Módulo de comunicação (CM)	RS232	Full-duplex
	RS485	Half-duplex
	RS422/485	Full-duplex (RS422) Half-duplex (RS485)
	PROFIBUS Mestre	DPV1
	PROFIBUS Escravo	DPV1
	AS-I Mestre (CM 1243-2)	AS-Interface
① Processador de comunicação (CP)	Conectividade	GPRS
① Placa de comunicação (CB)	RS485	Half-duplex
① Teleserviço	Adaptador TS IE Basic	Conexão ao CPU
	Adaptador TS GSM	GSM/GPRS
	Adaptador TS Modem	Modem
	Adaptador TS ISDN	ISDN
	Adaptador TS RS232	RS232

3.1.3. Introdução ao *software* de programação

O *software* utilizado para a programação do PLC e construção do ambiente gráfico e menus da consola HMI foi o STEP 7 V11 UPDATE 2. Este *software* fornece um ambiente amigável para o utilizador (*user-friendly*) criar projectos, converter projectos criados em programas de versões anteriores e efectuar migrações. Além disso, permite a total configuração, programação, controlo e monitorização de vários PLC's quando ligados em rede.

O seu ambiente gráfico bastante melhorado e mais limpo comparativamente a versões anteriores do programa, permite uma melhor, mais simples, eficiente e rápida criação de novos projectos.

A última versão deste *software* veio trazer bastantes melhorias face a versões anteriores [27]:

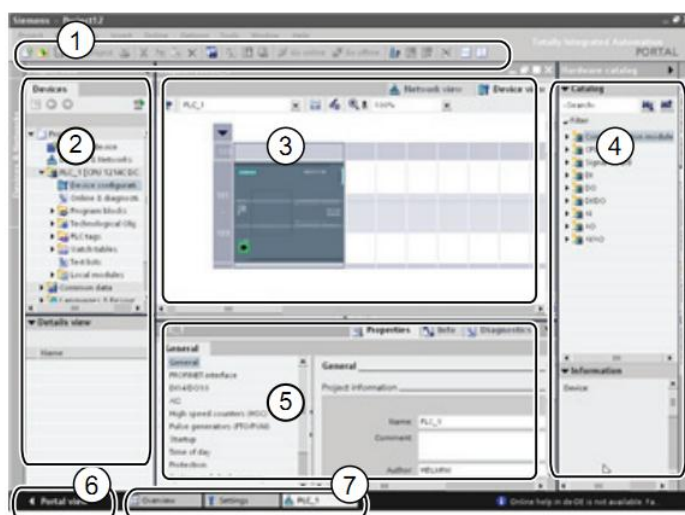
- Permite um maior controlo na criação de variáveis e providencia tipos adicionais de informação, tais como ponteiros, matrizes indexadas e estruturas;
- Efectua automaticamente e implicitamente conversões de dados de pequenas dimensões (*SInt* ou *Byte*) para formatos de dados de maiores dimensões quando necessário (*DInt*, *DWord*, etc.);
- Permite efectuar alterações ao programa do PLC sem ser necessário colocar o autómato no modo parado;
- Possibilita a comunicação PROFINET UDP - UDP permite transmissão (*broadcast*) de comunicações;
- Implementação da função “undo” (desfazer ou voltar atrás - Ctrl + Z);
- Entre outras.

Possui também a vantagem de permitir a programação e visualização do programa em 3 tipos distintos de linguagem: *ladder*, FBD e SCL [27].



- ① Portais para diferentes tarefas
- ② Tarefas para o portal seleccionado
- ③ Painel de selecção para a acção seleccionada
- ④ Vai para a vista do projecto

Figura 3.7 - Janela inicial do STEP 7 V11 [27].



- ① Menus e barras de ferramentas
- ② Navegador do projecto
- ③ Área de trabalho
- ④ Tarefas, instruções e funções
- ⑤ Janela de inspecção
- ⑥ Vai para a vista do portal
- ⑦ Barra de selecção de janela

Figura 3.8 - Janela de projecto do STEP 7 V11 [27].

Graças à sua intuitiva e simples *interface*, criar um programa ou projecto é relativamente simples para quem possui o mínimo de conhecimento da linguagem de programação em questão.

Basta arrastar a instrução ou função desejada para a *rung*



Figura 3.9 - Facilidade na criação de programas.

3.1.4. Introdução à consola KTP600 Basic Mono PN

Para uma melhor visualização, controlo e monitorização do sistema foi utilizada uma consola HMI, mais especificamente a KTP600 Basic Mono PN da Siemens. Esta consola apresenta um ecrã LCD monocromático de 5.7", 320 x 240 pixéis de resolução e ainda a possibilidade de controlo do contraste [29].

Além disso, possui um tipo de entrada táctil e 6 teclas físicas de funções. Permite o armazenamento de um programa de tamanho máximo de 512 *KBytes* e comunicação RJ45 (*ethernet*).

Relativamente à sua alimentação, deve ser utilizada uma fonte DC de 24V, podendo a tensão de entrada variar entre 19.2 e 28.8V.

Possui também a capacidade de sincronização do relógio interno através de *software* [29].

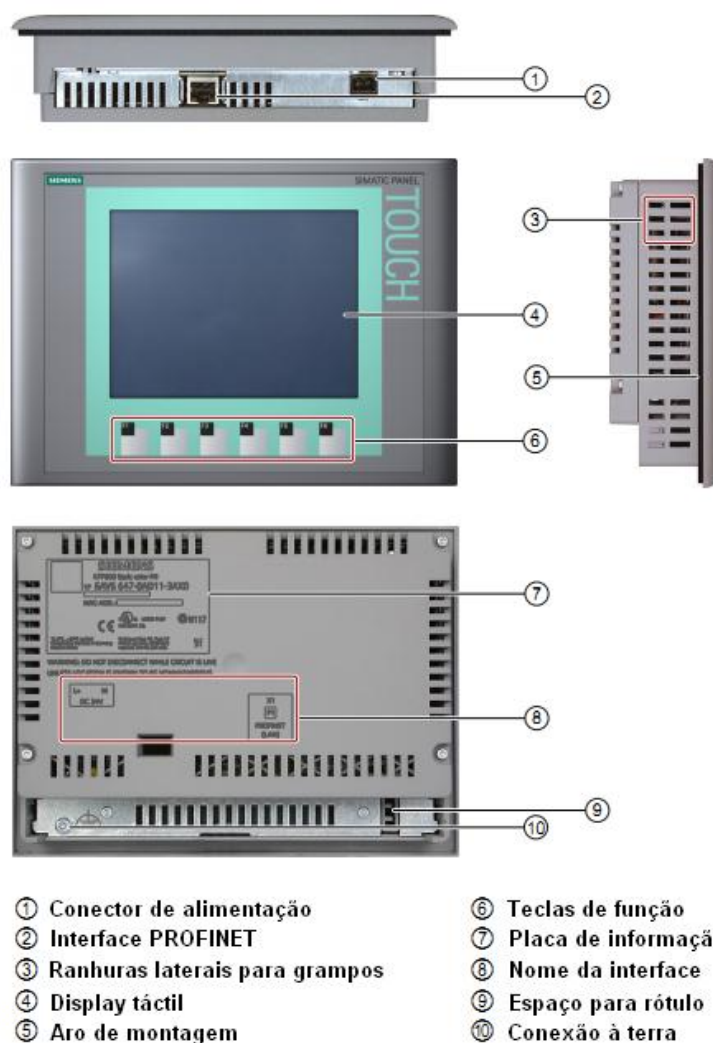


Figura 3.10 - Constituição da consola HMI [29].

3.1.5. Detalhes do sistema

O sistema, como referido anteriormente, permite o controlo e monitorização, quer local, quer remota, de um sistema de rega inteligente.

Através dos seus 4 modos de funcionamento, garante-se o controlo do sistema, impedindo que, em qualquer situação, o tanque fique sem água.

O funcionamento geral do sistema consiste no seguinte: uma bomba de água capta água de um furo, poço ou ponto de captação e armazena-a num reservatório de 3000 litros. Existe um sensor na bomba que detecta a presença de água na bomba. Caso o sensor não detecte água na bomba, a bomba de água pára de funcionar, evitando assim qualquer dano mecânico causado pelo funcionamento em seco da bomba.

O pH da água do reservatório está constantemente a ser analisado por uma sonda de pH (para efeitos de simulação, foi utilizado um potenciómetro que simula os valores entre 0 e 14 do pH). Quando o valor do pH da água se encontra abaixo do valor de set-point do pH definido pelo utilizador (valor predefinido: 4), a bomba do pH é ligada e começa a ser debitado um reagente químico para contrariar a redução do pH. Se por algum motivo o valor do pH baixar do valor de alarme do pH definido pelo utilizador (set-point de alarme - valor predefinido: 2), o fornecimento de água é imediatamente cortado (a válvula de água é fechada). A velocidade do débito do reagente químico é proporcional à diferença entre o valor de set-point do pH e valor de alerta do pH, ou seja, quanto maior for a diferença entre o valor do set-point do pH e o valor de alerta do pH, maior vai ser a velocidade de débito da bomba. Este sistema está também preparado para a utilização de um agitador mecânico, de modo a homogeneizar o reagente químico por toda a água do reservatório, evitando assim débitos imprecisos do reagente.

No modo 1 de funcionamento (Manual), o sistema permite:

- Ligar/desligar a bomba de água;
- Ligar/desligar a bomba do pH;
- Abrir/fechar a válvula de água;
- Alteração da velocidade da bomba do pH (0 a 100 % - velocidade mínima e velocidade máxima, respectivamente).

No modo 2 de funcionamento (Automático por nível), o sistema é automaticamente controlado de acordo com o valor do set-point de água definido pelo utilizador (valor predefinido: 1500 litros). A bomba de água é ligada quando o nível do tanque atingir o valor do set-point e só desliga quando o tanque estiver cheio.

Por sua vez, o modo 3 de funcionamento (Automático por diferencial) é bastante semelhante ao modo 2. Contudo, apresenta mais um valor que permite uma redução no tempo de funcionamento da bomba de água, o valor do set-point diferencial.

Este valor (valor predefinido: 500) actua em conjunto com o valor do set-point de água definido no modo 2, e funciona da seguinte forma:

- A bomba é ligada quando o nível da água atingir o valor de set-point - diferencial (SP - D), que neste caso corresponde a $1500 - 500 = 1000$ litros;
- A bomba é desligada quando o nível de água atingir o valor de set-point + diferencial (SP + D), que neste caso corresponde a $1500 + 500 = 2000$ litros.

Assim, a bomba apenas irá trabalhar entre os 1000 e 2000 litros de água, evitando que o sistema esteja constantemente a encher o depósito até ao limite máximo (modo 2).

Por último, o modo 4 (Modo nocturno) permite uma melhor gestão e poupança de energia, pois permite que a bomba de água trabalhe apenas durante o período nocturno (das 24h às 6h), período em que o valor da electricidade é mais barato (para contadores bi-horários). Durante o restante período (período diurno) a bomba não trabalha. Este modo é aconselhável para quando se pretende regar apenas durante a noite. Se por algum motivo o utilizador pretender regar durante o dia e se esquecer de alterar o modo de funcionamento, quando o nível mínimo do tanque é atingido, o sistema passa automaticamente para o modo 2 de funcionamento, evitando que o tanque fique vazio em qualquer situação.

Em qualquer um dos modos, caso o valor do pH desça abaixo do valor de alerta definido pelo utilizador, o fornecimento de água é imediatamente cortado. Em caso de falha dos sensores de nível, ou caso haja alguma ruptura no reservatório e o débito da bomba não consiga acompanhar o débito do gasto de água e o tanque ultrapasse o nível mínimo de água, o fornecimento de água é automaticamente cortado, evitando assim a desferragem dos circuitos de rega. O sistema apresenta 5 circuitos de rega que podem ser controlados individualmente e a qualquer instante a partir da consola HMI ou da página *web* (ver Anexo I e II).

O sistema possui ainda uma funcionalidade que simula as horas necessárias restantes para manutenção da bomba (valor predefinido: 10000 horas). Quando a bomba tiver funcionado 9952 horas ($10000 - 48 = 9952$ horas), será acesa uma luz de aviso de manutenção.

NOTA: O nível mínimo de água é definido de modo a que permaneçam sempre cerca de 200 litros de água no tanque em qualquer momento.

3.1.5.1. Programa de controlo

Para o controlo, monitorização e implementação do sistema, foi desenvolvido um programa com recurso ao já apresentado *software* STEP 7 V11.

Este programa é responsável por todas as decisões tomadas pelo PLC; envia comandos de controlo (através das saídas digitais analógicas e digitais) aos diversos componentes do sistema de acordo com os dados e informações que recebe dos sensores (entradas analógicas e digitais).

Antes de se iniciar a criação do programa de controlo, foi necessário pré-configurar alguns parâmetros do PLC e estabelecer algumas regras:

- Adição dos respectivos módulos adicionais ao PLC, neste caso, a placa de uma saída analógica incorporada (AQ1 x 12 bits);

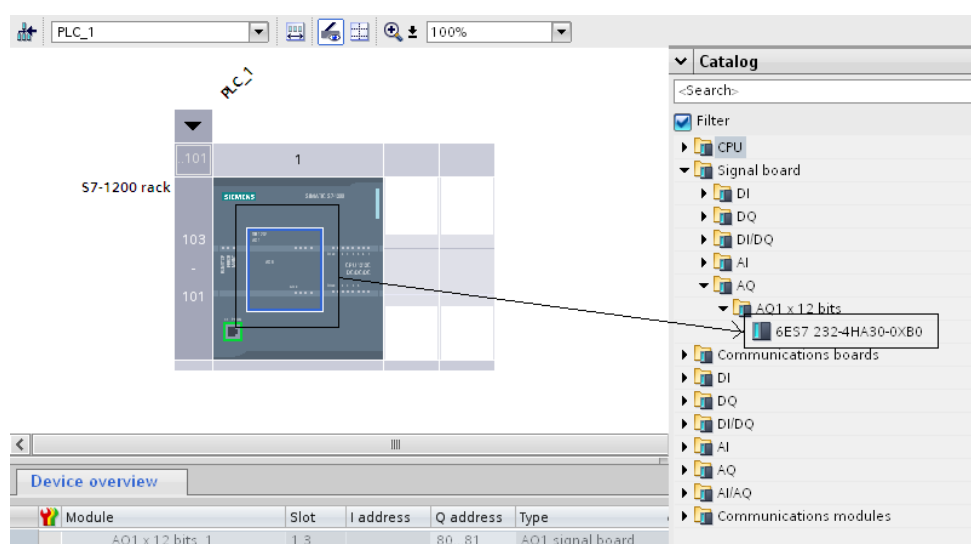


Figura 3.11 - Módulo de uma saída analógica.

- Atribuição do endereço de IP do PLC;

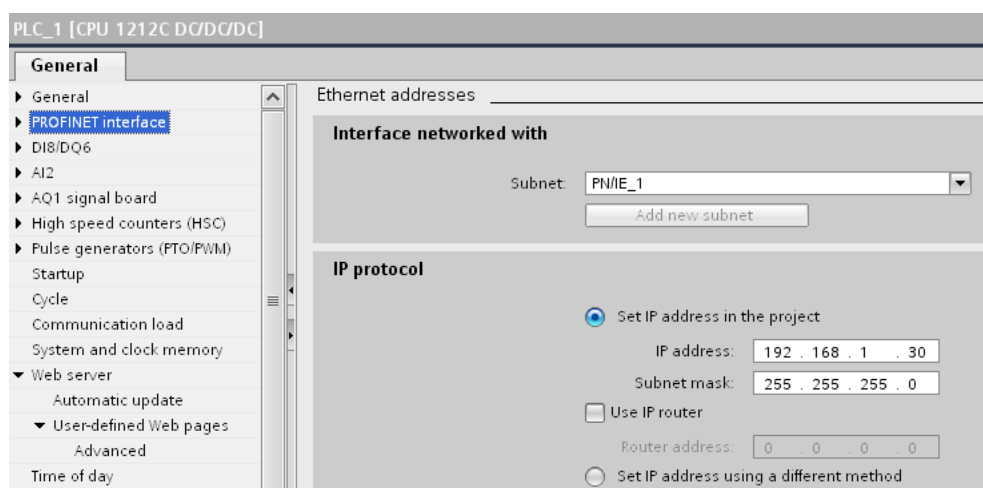


Figura 3.12 - Atribuição do endereço de IP ao PLC.

- Activação dos *bits* pré-configurados do sistema e relógio de memória;

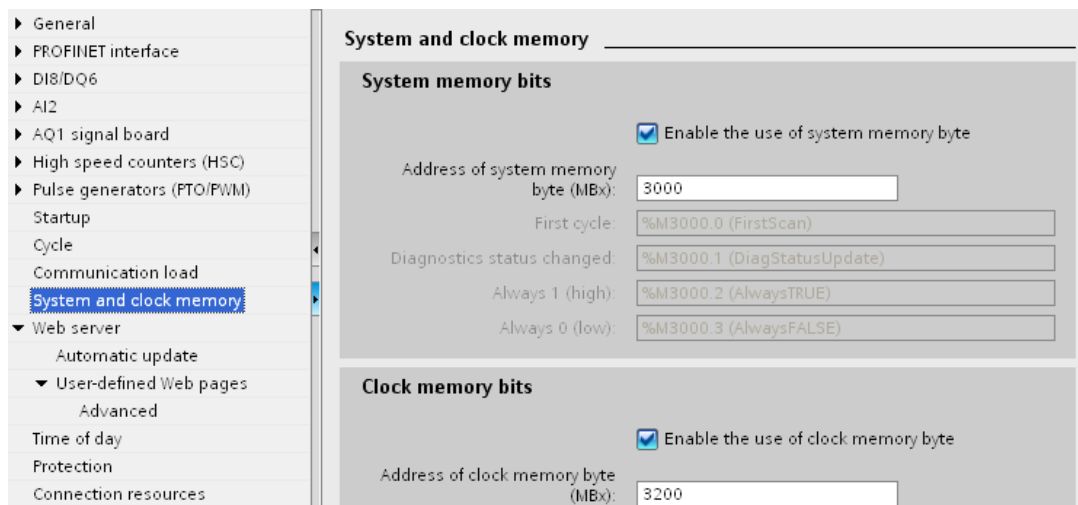


Figura 3.13 - Activação de *bits* do sistema.

- Activação do sistema *Web Server* do PLC e respectivas configurações;

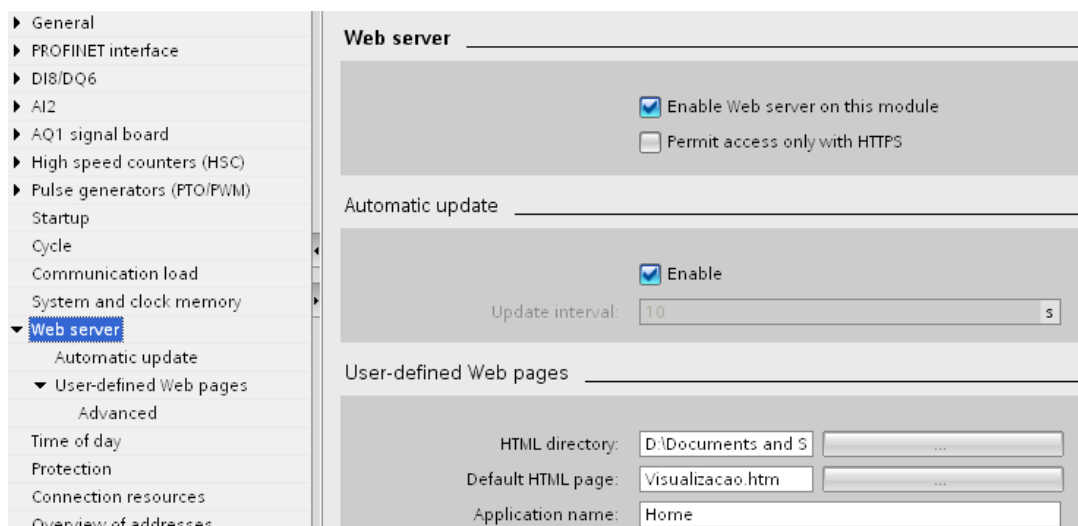


Figura 3.14 - Activação do sistema *Web Server*.

- Outras configurações adicionais necessárias ao correcto funcionamento e programação do PLC.

De seguida, foi iniciada a criação do programa de controlo. A estrutura de programação e organização do programa foram implementadas de acordo com a execução lógica do programa, ou seja, visto que o funcionamento do PLC consiste na constante realização do processo de *scan* (devidamente explicado no Capítulo 2) num *loop* (ciclo) infinito, as configurações e partes do programa mais importantes foram implementadas de acordo com a sua prioridade, sendo que as mais prioritárias foram colocadas no início do programa, seguindo-se as restantes de forma decrescente. Foram também escritos

comentários referentes a cada *network* (ou secção) do programa para uma melhor e mais fácil leitura e compreensão do programa.

Inicialmente, começou por ser implementada a criação da página *web* do PLC. Nesta *network* estão presentes todas as instruções e variáveis referentes à inicialização e funcionamento da página *web*.

De seguida, implementaram-se as condições de arranque e paragem do motor da bomba de água. Esta *network* consiste basicamente no seguinte:

- Quando o *bit* do sistema (*bit* que indica se o sistema está activado ou desactivado - menu “Arranque e Funcionamento” da consola HMI e página “Controlo Manual” da página *web*, Anexos I e II respectivamente) está a 1 (activado) e caso:
 - ✓ O modo 1 de funcionamento esteja seleccionado e o *bit* referente ao controlo manual da bomba de água esteja activo;
- OU
- O modo 2 de funcionamento esteja seleccionado e o *bit* auxiliar (que indica se o sistema está a encher ou está parado) esteja activo;
- OU
- O modo 3 de funcionamento esteja seleccionado e o *bit* auxiliar esteja activo;
- OU
- O modo 4 de funcionamento está seleccionado, o *bit* do modo nocturno esteja activado, assim como o *bit* auxiliar;
- E
- O nível máximo não tenha sido atingido;

→ O motor da bomba de água é ligado.

Na *network* seguinte, foram implementadas as condições que implicam o funcionamento ou paragem da bomba de água.

- Caso o *bit* do sistema esteja activado, o modo de funcionamento seleccionado seja entre 1 e 4, o *bit* do motor (estado deste *bit* foi definido na *network* anterior) esteja activado, o sensor da bomba detecte água e o botão de emergência não tenha sido pressionado, **a bomba é activada.**

Na *network* 4 é tomada a decisão de abrir ou fechar a válvula de distribuição de água para os circuitos de rega. Esta decisão assenta nos seguintes factores:

- Caso o *bit* do sistema esteja activado, o modo de funcionamento seleccionado seja o manual e o *bit* de controlo manual da válvula de água esteja activado;

OU

- Caso o *bit* do sistema esteja activado e o modo de funcionamento seleccionado seja entre 2 e 4;

E

- O *bit* referente ao valor de alarme do pH não esteja activado, o nível mínimo não tenha sido atingido nem o botão de paragem de emergência pressionado;

→ **A válvula de distribuição de água é aberta.**

Posteriormente, segue-se a implementação do controlo dos circuitos de rega.

Na *network* 6, 7 e 8 são implementadas as condições que definem a activação dos *bits* referentes a modo de funcionamento 2, 3 e 4, respectivamente.

Na *network* seguinte é implementado o sistema que simula a quantidade de água fornecida através de um caudalímetro. A cada 10 impulsos do caudalímetro, é contabilizado 1 m³ de água às contagens parcial e total. O *reset* da contagem parcial é implementado na *network* seguinte.

A *network* 11 é responsável pela apresentação do estado de funcionamento do sistema quer no ecrã da consola HMI quer na página *web*. As opções são as seguintes: Encher, Parado, Cheio/Parado ou Vazio/Encher.

Da *network* 12 até à 14, é implementado o relógio interno do sistema e também o sistema que permite a contabilização do tempo de funcionamento total e parcial da bomba de água. Na *network* 15 é implementada a possibilidade de fazer *reset* ao contador parcial de horas de funcionamento da bomba.

De seguida seguem-se duas *networks* que são responsáveis pela contagem do tempo restante para manutenção da bomba de água, activação do respectivo alarme de aviso de manutenção e *reset* do mesmo.

Da *network* 18 à 24 são implementadas todas as instruções, condições e conversões necessárias referentes aos vários valores do pH (set-point, valor de alerta e alarme do pH), apresentação do valor de pH nos diversos locais de visualização do sistema, activação ou paragem da bomba doseadora do pH e controlo automático da velocidade dessa mesma bomba.

Seguem-se as implementações referentes à conversão do valor da quantidade de água no tanque para m³ e litros, assim como a activação e paragem do agitador mecânico, respectivas indicações de tanque cheio ou vazio e ainda as sinalizações (alarmes) relativas à paragem de emergência do sistema, falta de água na bomba, valor de alarme do pH atingido e manutenção da bomba de água.

Na *network* 30 foi implementado um sistema que simula o ano bissexto (Fevereiro com 29 dias). Este sistema é muito importante na criação dos *data logs*, pois evita eventuais erros na criação dos mesmos relativamente ao tamanho do *data log* do mês de Fevereiro. Caso este sistema não fosse implementado, no decorrer de um ano bissexto, os erros propagar-se-iam pela criação dos *data logs* referentes aos restantes meses do ano.

Da *network* 31 até à 37 é implementada a criação de *data logs* do sistema. Esta implementação será analisada com maior detalhe mais à frente nesta dissertação.

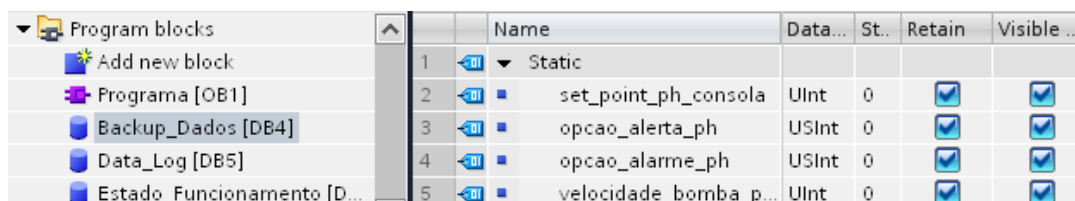
Seguidamente, é implementado um método para evitar erros de introdução relativamente ao modo de funcionamento do sistema na consola HMI.

Na *network* 38 e 39 estão presentes as instruções referentes à reposição dos valores de origem do sistema e também a indicação do estado de funcionamento geral do sistema (Ligado ou Desligado).

Na última *network* do programa foram implementadas duas instruções que permitem o ajuste dos valores das entradas analógicas do PLC. Este ajuste é necessário efectuar apenas aquando da alteração da alimentação do PLC, pois tal alteração vai influenciar os valores de corrente debitados ao PLC pela variação dos potenciómetros referentes à simulação da quantidade de água actual do tanque e valor do pH da água.

Ao longo da criação do programa, foram criadas diversas variáveis e atribuídos diversos endereços de memória necessários ao correcto funcionamento do programa.

Para evitar que todos os dados e valores das variáveis do programa fossem perdidos aquando de uma falha de alimentação do PLC, foram utilizadas *data blocks* (blocos de dados) com a função *retain* (reter). Esta função faz com que os valores das variáveis utilizadas nos blocos de dados sejam guardadas directamente na memória não volátil do PLC, e portanto, não são afectados por eventuais quebras na alimentação do PLC.



	Name	Data...	St..	Retain	Visible ...
1	Static				
2	set_point_ph_consola	UInt	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
3	opcao_alerta_ph	USInt	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
4	opcao_alarme_ph	USInt	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
5	velocidade_bomba_p...	UInt	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Figura 3.15 - Função *retain* dos blocos de dados.

3.1.5.2. Implementação da base de dados

Como referido anteriormente, foi também implementado um sistema que permite o registo e armazenamento de dados relativos ao sistema, nomeadamente a quantidade de água consumida (m^3), o tempo total de funcionamento da bomba de água (dias, horas e minutos), os litros de água presentes no tanque e o valor de pH no momento do registo. Para complementar a informação, cada registo tem associado o nº de registo e a data e hora a que foi efectuado (*timestamp*).

Este sistema foi implementado com recurso a uma funcionalidade interna do PLC utilizado neste projecto.

O seu funcionamento consiste no seguinte:

- No início de cada mês, às 00:00:00 horas é criado e aberto um *data log* (Fig 3.16);

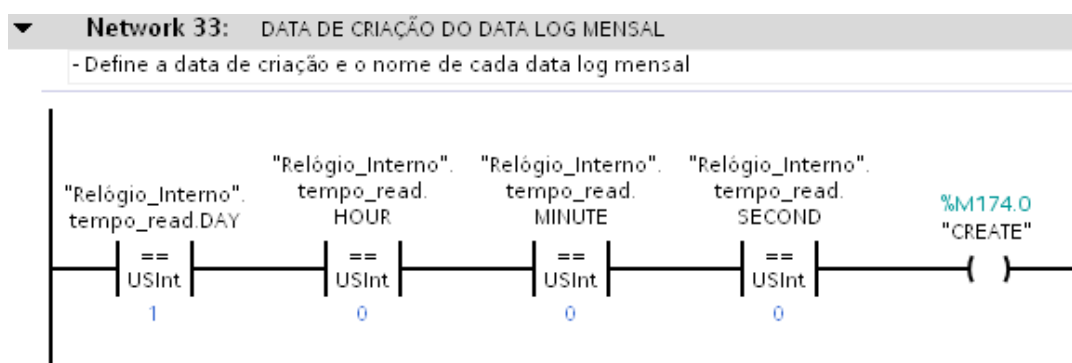


Figura 3.16 - Criação e abertura de um *data log*.

- Todos os dias às 23:59:00 horas de cada dia é realizado um registro (Fig. 3.17);

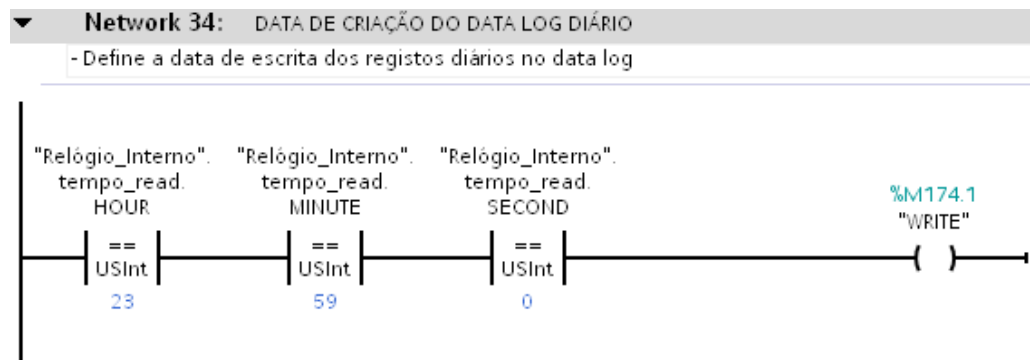


Figura 3.17 - Criação do registro diário.

- No dia final de cada mês, mais concretamente às 23:59:05 horas desse dia ou quando o registro desse mês estiver cheio (mecanismo utilizado para evitar falhas no fecho de cada *data log* devido a eventuais falhas de corrente no PLC, obrigando assim que o *data log* seja fechado quer esteja cheio quer seja o fim de cada mês), o *data log* correspondente a esse mês é fechado (Fig. 3.18);

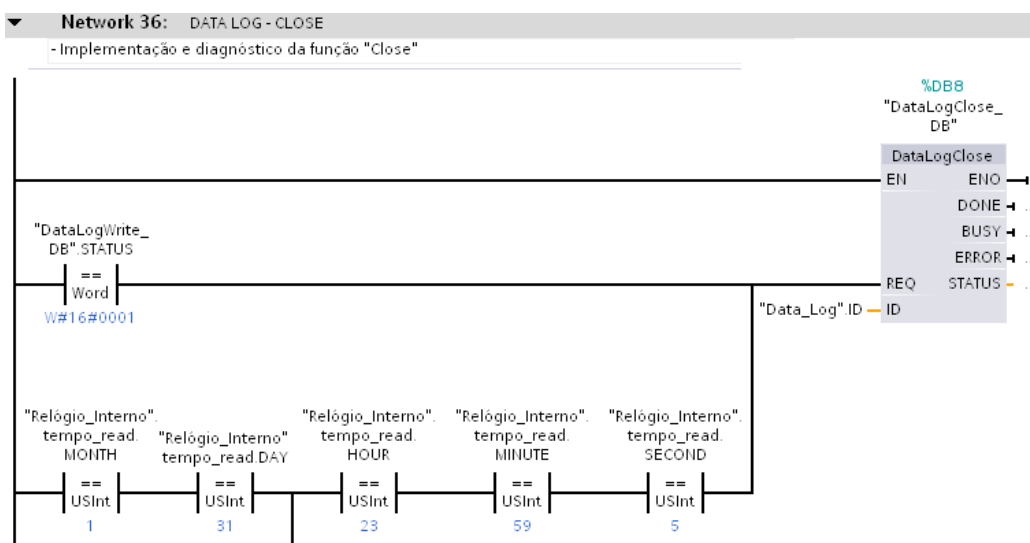


Figura 3.18 - Fecho do *data log* mensal.

- O processo recomeça novamente e repete-se sucessivamente.

NOTA: Cada vez que o PLC for reiniciado (p. ex. falha na alimentação), o *data log* do mês em questão é aberto para possibilitar a continuação do registro diário (Fig. 3.19). Caso contrário, o registro diário não poderia ser realizado até à criação do *data log* referente ao mês seguinte.

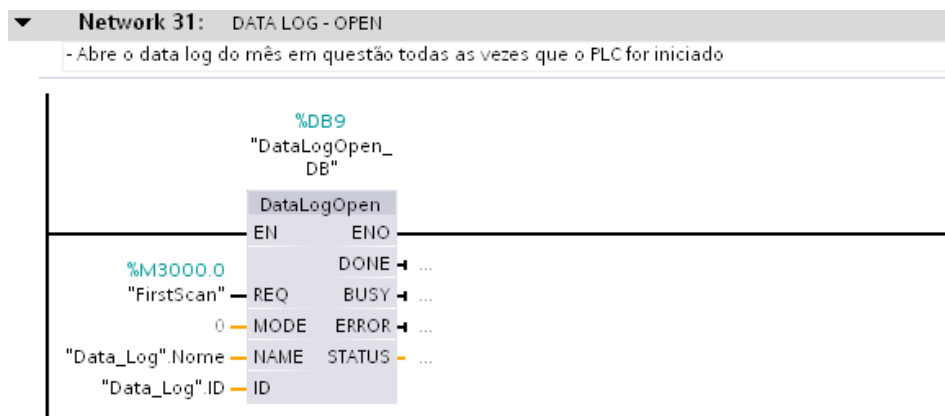


Figura 3.19 - Abertura do *data log* cada vez que o PLC é reiniciado.

O nome dos *data logs* é atribuído consoante o mês em questão.

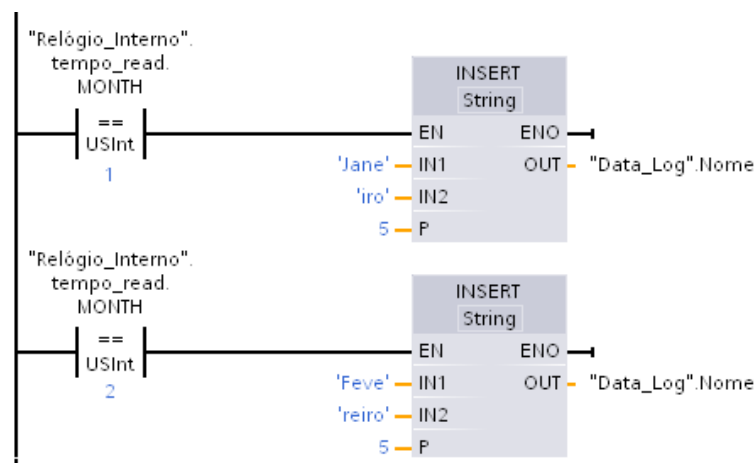


Figura 3.20 - Atribuição do nome aos *data logs*.

Como referido anteriormente, foi implementado um sistema para determinar se o ano é ou não bissexto.



Figura 3.21 - Determinação de ano bissexto.

O utilizador deve apenas ter o cuidado, por exemplo, de meio em meio ano ou 3 em 3 meses descarregar e apagar os *data logs* da memória interna do PLC, pois o PLC possui memória finita e quando chega ao fim de um ano de armazenamento de dados ou a sua memória interna fica cheia, começa a sobrepor os valores registados nos *data logs* do ano anterior.

O processo para consultar, aceder e apagar os *data logs* já foi referido em cima (através da *interface Web Server* do PLC).

Em baixo é apresentado o conteúdo de um *data log* criado neste sistema.

	A	B	C	D	E	F	G	H	I
1	Record	Date	UTC Time	Água Consumida (m3)	Dias	Horas	Minutos	Litros no Tanque	Valor do PH
2	1	1/01/2012	23:59:00	16	0	2	14	1204	14
3	2	1/02/2012	23:59:00	16	0	2	14	1205	13
4	3	1/03/2012	23:59:00	16	0	2	14	1203	14
5	4	1/04/2012	23:59:00	16	0	2	14	1204	13
6	5	1/05/2012	23:59:00	16	0	2	14	1204	14
7	6	1/06/2012	23:59:00	16	0	2	14	1203	14
8	7	1/07/2012	23:59:00	16	0	2	14	1204	14
9	8	1/08/2012	23:59:00	16	0	2	15	1204	14
10	9	1/09/2012	23:59:00	16	0	2	15	0	14
11	10	1/10/2012	23:59:00	16	0	2	15	0	14
12	11	1/11/2012	23:59:00	16	0	2	15	0	9
13	12	1/12/2012	23:59:00	16	0	2	15	0	9
14	13	1/13/2012	23:59:00	19	0	2	15	0	9
15	14	1/14/2012	23:59:00	19	0	2	15	0	0
16	15	1/15/2012	23:59:00	19	0	2	15	3000	0
17	16	1/16/2012	23:59:00	19	0	2	15	0	8
18	17	1/16/2012	23:59:00	20	0	2	15	0	8
19	18	1/18/2012	23:59:00	20	0	2	16	0	8
20	19	1/19/2012	23:59:00	20	0	2	16	0	8
21	20	1/20/2012	23:59:00	20	0	2	16	0	8
22	21	1/21/2012	23:59:00	22	0	2	16	0	8
23	22	1/22/2012	23:59:00	22	0	2	16	2222	8
24	23	1/23/2012	23:59:00	22	0	2	16	2220	2
25	24	1/24/2012	23:59:00	22	0	2	16	2220	2
26	25	1/25/2012	23:59:00	22	0	2	16	2221	2
27	26	1/26/2012	23:59:00	22	0	2	16	2220	2
28	27	1/27/2012	23:59:00	22	0	2	16	2220	2
29	28	1/28/2012	23:59:00	22	0	2	17	2221	14
30	29	1/29/2012	23:59:00	22	0	2	17	2222	13
31	30	1/30/2012	23:59:00	23	0	2	17	2222	14
32	31	1/31/2012	23:59:00	23	0	2	17	2222	13

Figura 3.22 - Exemplo devidamente organizado de um *data log* do mês de Janeiro.

As vantagens de possuir um sistema de registo e armazenamento de dados são as seguintes:

- Possibilidade de comparar dados entre dias, meses e mesmo anos para fins estatísticos;

- Possibilidade de analisar dados para detecção de eventuais falhas (p. ex. valores de pH muito baixos) que tenham ocorrido;
- Possibilidade de analisar dados para saber quais os dias ou meses em que se gastou mais água, em que a bomba trabalhou mais tempo e durante que período, valores do pH diários, etc.

3.1.5.3. Programa da consola HMI

Para uma mais completa implementação, controlo e monitorização do sistema, foi implementado um programa na anteriormente referida consola HMI.

O programa da consola está dividido por menus de configuração e visualização e qualquer alteração que seja feita, quer seja alterar o set-point da água, do pH, alterar o modo de funcionamento, ligar/desligar todo o sistema ou alterar o relógio interno do sistema necessita que o utilizador esteja identificado como administrador do dispositivo (*login* de administrador). Caso contrário, nenhuma alteração poderá ser efectuada.

O *login* de administrador é automaticamente terminado após 2 minutos de inutilização da consola, evitando assim o esquecimento de término da sessão de administrador (*logout*) e a possibilidade de qualquer pessoa alterar as configurações do sistema.

Para a implementação do programa da consola, recorreu-se novamente ao programa STEP 7 V 11 UPDATE 2.

Inicialmente, foi necessário adicionar a consola HMI ao já existente projecto do PLC.

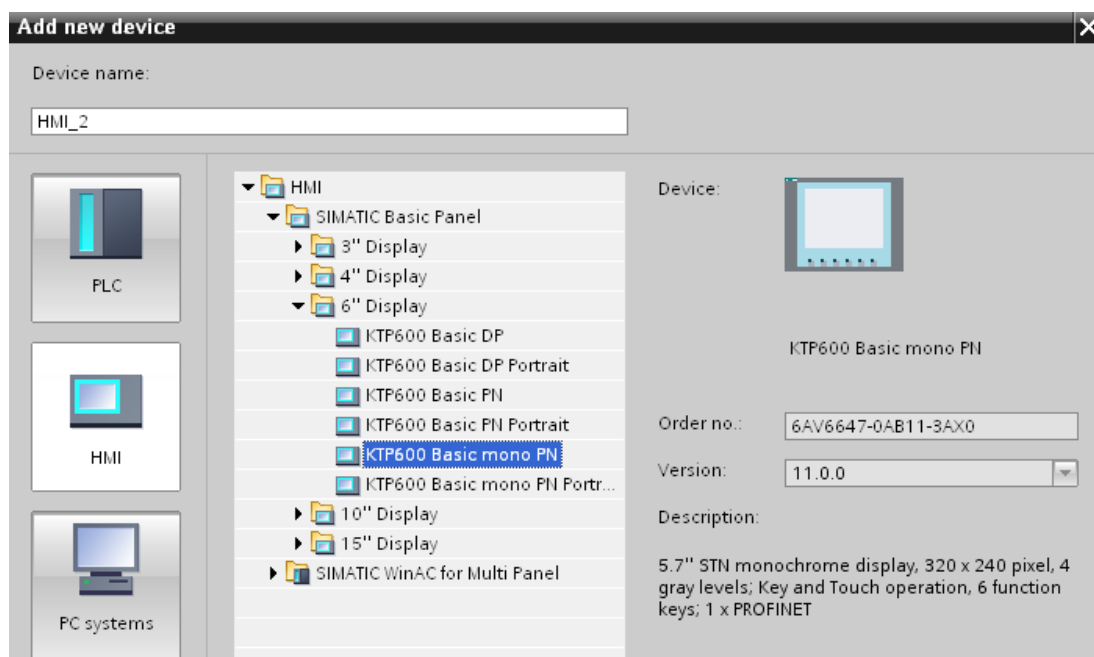


Figura 3.23 - Adição da consola HMI ao projecto.

De seguida, foi necessário configurar os parâmetros e o endereço de IP da consola.

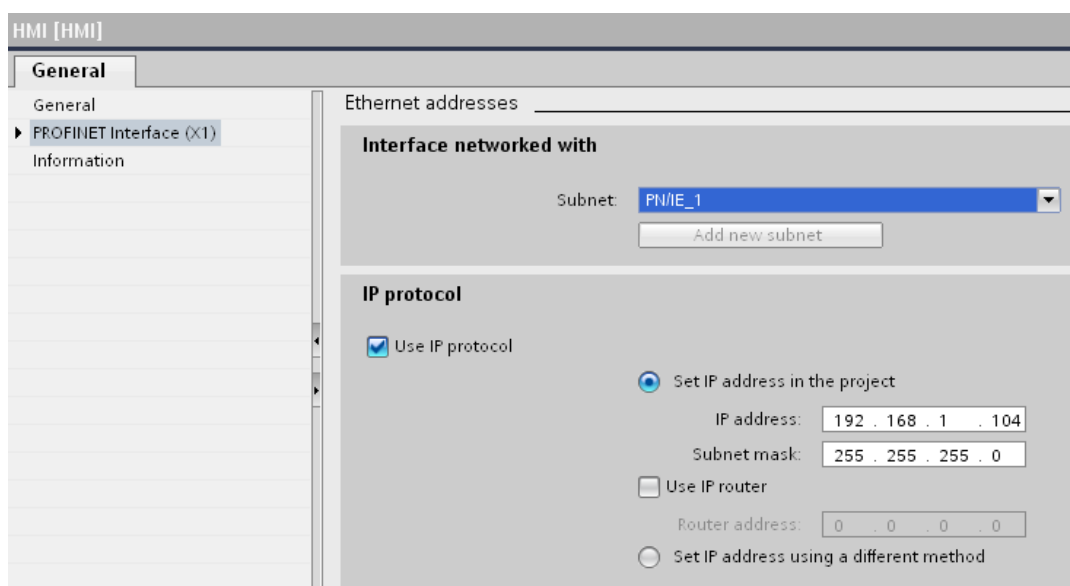


Figura 3.24 - Configuração do IP e *network* da consola.

Por último, configurou-se a interligação entre o PLC e a consola HMI.

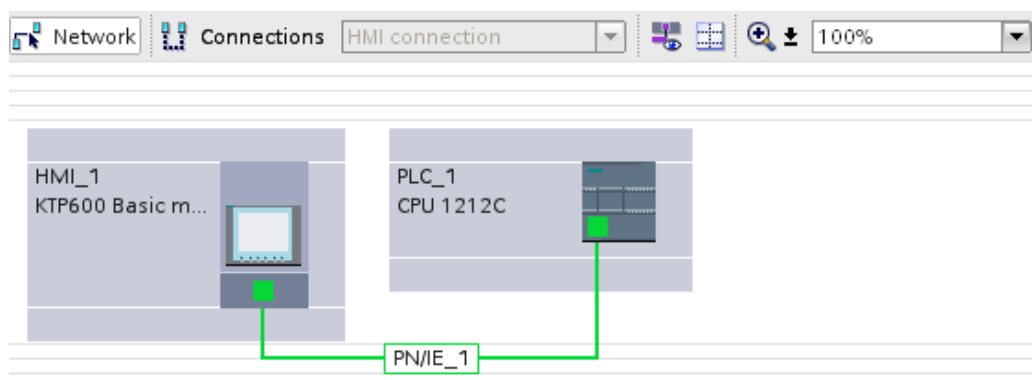


Figura 3.25 - Configuração da ligação entre PLC e consola HMI.

Posteriormente, iniciou-se a criação do programa. O primeiro passo foi criar os ecrãs necessários e configurar os *templates* (modelos) a utilizar. Após isso, criou-se o *login* de administrador e configuraram-se as devidas permissões. Foi também criada a lista de entradas referente aos nomes dos diversos menus.

A listagem e estruturação dos menus seguiu a mesma base que a do programa do PLC: ser fácil de interpretar e o mais acessível possível e intuitivo para o utilizador.

Os vários menus constituintes do programa da consola são devidamente apresentados e explicados no Anexo I desta dissertação.

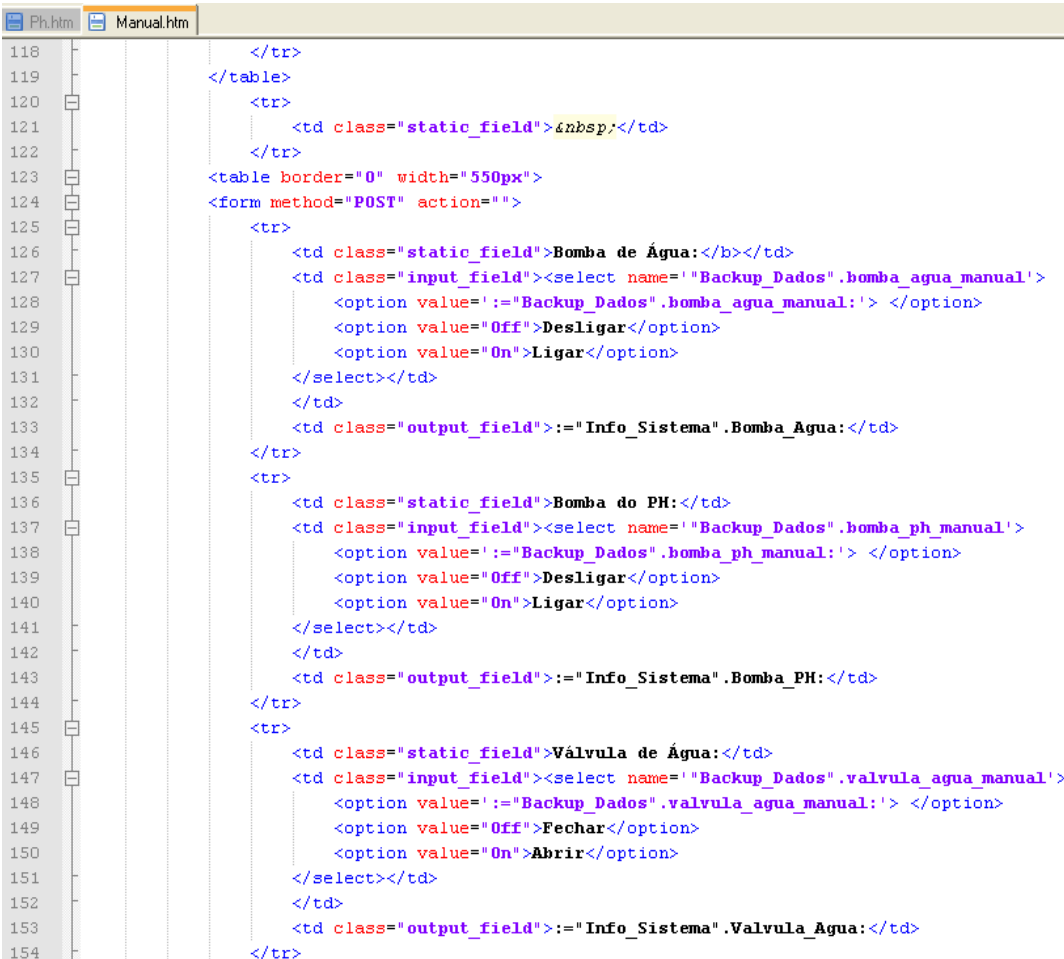
3.1.5.4. Criação da página web

De forma a complementar o sistema, foi também criada uma página web interna do PLC que permite monitorizar e controlar todos os componentes constituintes do sistema. Esta página, tal como a consola HMI, requer que o utilizador esteja identificado como administrador para conseguir alterar qualquer que seja o parâmetro do sistema. Caso contrário, é-lhe permitido apenas visualizar o sistema.

Para a criação de uma página web, foi necessário activar a função *Web Server* do PLC (Fig. 3.14).

Após a compilação do programa, o PLC cria automaticamente o ficheiro referente à página inicial do utilizador.

Para a criação das diversas páginas de monitorização e controlo do sistema foi utilizado o programa *Notepad++*. Este potente e gratuito programa permite alterar como ficheiro de texto, vários tipos de formatos e linguagens de programação, incluindo HTML (formato utilizado na criação das páginas web deste sistema).



```
118      </tr>
119    </table>
120    <tr>
121      <td class="static_field">&nbsp;</td>
122    </tr>
123    <table border="0" width="550px">
124      <form method="POST" action="">
125        <tr>
126          <td class="static_field">Bomba de Água:</b></td>
127          <td class="input_field"><select name=' "Backup_Dados".bomba_agua_manual'>
128            <option value=' :="Backup_Dados".bomba_agua_manual:'> </option>
129            <option value=" Off">Desligar</option>
130            <option value=" On">Ligar</option>
131          </select></td>
132        </tr>
133        <td class="output_field">:="Info_Sistema".Bomba_Agua:</td>
134      </tr>
135      <tr>
136        <td class="static_field">Bomba do PH:</td>
137        <td class="input_field"><select name=' "Backup_Dados".bomba_ph_manual'>
138          <option value=' :="Backup_Dados".bomba_ph_manual:'> </option>
139          <option value=" Off">Desligar</option>
140          <option value=" On">Ligar</option>
141        </select></td>
142        </td>
143        <td class="output_field">:="Info_Sistema".Bomba_PH:</td>
144      </tr>
145      <tr>
146        <td class="static_field">Válvula de Água:</td>
147        <td class="input_field"><select name=' "Backup_Dados".valvula_agua_manual'>
148          <option value=' :="Backup_Dados".valvula_agua_manual:'> </option>
149          <option value=" Off">Fechar</option>
150          <option value=" On">Abrir</option>
151        </select></td>
152        </td>
153        <td class="output_field">:="Info_Sistema".Valvula_Agua:</td>
154      </tr>
```

Figura 3.26 - Interface de edição do programa *Notepad++*.

Assim, e para a implementação das páginas web deste projecto, foram criadas 8 páginas independentes.

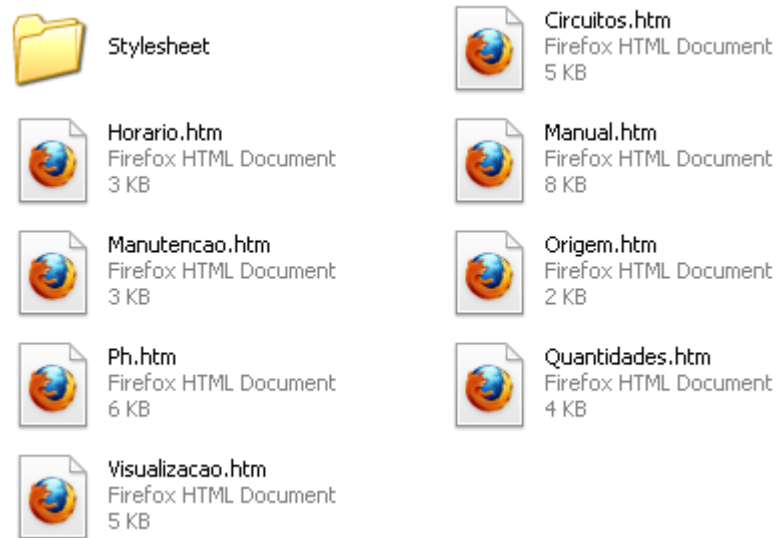


Figura 3.27 - Páginas web criadas.

A interligação entre as diversas páginas foi feita dentro de cada uma delas, ou seja, a página completa do sistema é apenas uma mas possui várias “subpáginas” no seu interior (ver Anexo II).

Para a implementação das páginas recorreu-se às mais diversas instruções e comandos HTML, tais como:

```
<tr>
<td class="static_field">Circuito #4:</b></td>
<td class="input_field"><select name="Backup_Dados".circuito_4'>
<option value=":"Backup_Dados".circuito_4:'> </option>
<option value="Off">Desligado</option>
<option value="On">Ligado</option>
</select></td>
<td class="output_field">:"Info_Sistema".Circuito_4:</td>
</tr> → (implementação de um menu de escolhas, neste caso “Ligado” ou “Desligado”).
```

Foi também necessário fazer a interligação entre as variáveis do PLC e as variáveis utilizadas na leitura e escrita de valores através das páginas web com recurso aos seguintes comandos:

```
<!-- AWP_In_Variable Name="Backup_Dados".circuito_1' → (definição de variável de escrita);
```

<!-- AWP_Enum_Def Name="circuito1" Values=0:"Off",1:"On" → (definição de variável de enumeração);

<!-- AWP_Enum_Ref Name="Backup_Dados".circuito_1' Enum="circuito1" → (referência à variável de enumeração).

As maiores vantagens desta funcionalidade são óbvias: controlo e monitorização de todo o sistema à distância e possibilidade de descarregar os *data logs* da memória do PLC remotamente. As várias páginas *web* criadas estão presentes no Anexo II desta dissertação.

3.1.6. Potencialidades do sistema

3.1.6.1. Comunicação GSM/GPRS

Hoje em dia é bastante usual ver sistemas de recolha e monitorização de dados em tempo real. Estes, apresentam a possibilidade de serem controlados e monitorizados remotamente via GSM. Uma das eventuais futuras melhorias deste projecto seria a implementação de um módulo de comunicação GSM.

A comunicação GSM tem vindo a ser um dos mais confiáveis métodos de comunicação sem fios e é relativamente fácil de utilizar. Surgiu em 1982, quando a Conferência Europeia dos Administradores dos Correios e Telecomunicações (CEPT) criaram um Grupo Móvel Especial (significado original de GSM) com o objectivo de criar uma série de normas para o plano de comunicações futuro da União Europeia [30]. A comunicação GSM utiliza ambos FDMA e TDMA para multiplexar a comunicação GSM nas bandas de frequência [30].

A segurança nas redes GSM é providenciada pela pelos protocolos de codificação da família A5 [31].

A rede GSM providencia comunicação de dados *full-duplex*²⁴ [32]. Quando aliados a um PLC, os módulos de GSM permitem que um ou vários utilizadores o consigam controlar através de SMS e receber informações e alarmes no seu telemóvel provenientes do sistema que o PLC está a controlar, por exemplo, o utilizador envia uma SMS para ligar ou desligar um determinado componente do sistema e o sistema, quando configurado para tal, envia uma SMS para o utilizador com o estado e eventuais alarmes que possam ser activados.

O GSM, é baseado na multiplexagem TDMA e funciona bem para serviços de voz, pois os dados são transmitidos de forma síncrona [30].

Por sua vez, o esquema síncrono (TDMA) não é adequado à transmissão de dados. Para resolver este problema, foi criada a *General Packet Radio System* (GPRS). GPRS é um sistema que utiliza os operadores GSM para transmissão de dados. Permite velocidades de transmissão até 170Kbps [30].

²⁴ *full-duplex* - comunicação bidireccional; permite enviar e receber dados simultaneamente.

Assim, o sistema proposto para este projecto de controlo e monitorização de um sistema de rega inteligente assenta na comunicação GSM utilizando o sistema GPRS. O utilizador pode aceder à aplicação remotamente através de um comando enviado via SMS. Por sua vez, a SMS vai ser lida e decodificada pela CPU do PLC. O sistema deve para tal, incluir todos os componentes necessários para que todos os processos sejam cumpridos de forma rápida, eficiente e sem erros.

Tabela 3.5 - Funcionalidades de algumas redes sem fios existentes [30].

| Tecnologia | Banda (Ghz) | Taxa de <i>bits</i> máxima (Mbps) | Camada física | MAC | Distância aproximada |
|--------------|----------------|-----------------------------------|---------------|-------------|----------------------|
| GSM, GPRS | 0.9, 1.8 | 0.17 | GMSK | TDMA/TDD | 30 km |
| TETRA | 0.4 | 0.36 | DQPSK | TDMA | 50 km |
| IEEE 802.11b | 2.4 | 11 (DA), 2 (FH) | SS/DS-DQPSK | CSMA/CA | 200 m |
| IEEE 802.11a | 5 | 54 | OFDM | CSMA/CA | 200 m |
| HIPERLAN I | 5 | 20 | GMSK | EY-NPMA | Extensível |
| HIPERLAN II | 5 | 54 | OFDM | TDMA/TDD | Extensível |
| Bluetooth | 2.4 | 1 | SS/FH | TDMA/TDD | 10/100 m |
| IrDA | Infravermelhos | 16 | PPM | IrLAP/IrLAN | 1 m |

Para a implementação de um sistema GPRS neste projecto, seria necessário o seguinte equipamento (além do básico, que será um PLC e o *software* de programação) [33]:

- Processador de comunicação CP 1242-7;
- PLC com CPU e *firmware* igual ou superior à versão 2.0;
- Antena externa para o CP 1242-7 - ANT794-4MR ou ANT794-3M;
- PC com ligação à internet, para comunicação com o servidor central;
- Caso seja necessário utilizar o Teleserviço (permite descarregar o programa para o PLC e realizar diagnósticos) via GPRS, será também necessário uma *gateway* com acesso à internet. Esta *gateway* pode ser implementada com recurso a um PC com o *software* “TS Gateway” instalado.

Este sistema apresenta a vantagem de permitir aos utilizadores a possibilidade de controlarem e visualizarem um sistema remotamente, sem necessidade de estarem fisicamente presentes no local do mesmo.

Como apenas seria necessário, além do PLC, o processador de comunicação GSM CP 1242-7 e uma antena externa, pode-se concluir que este tipo de sistema é bastante acessível economicamente e simples de utilizar.

3.1.6.2. Monitorização do solo via *ZigBee* e GPRS

O sistema apresentado nesta dissertação está preparado para a implementação de um sistema de monitorização da temperatura, humidade e acidez do solo. Tal, possibilitaria um aumento na qualidade e quantidade de produção dos alimentos ou frutos plantados na área monitorizada.

Uma das formas de implementar um sistema deste tipo seria construir uma rede de infra-estruturas implementada com recurso a dispositivos *ZigBee* e GPRS.

ZigBee é a única tecnologia de redes sem fios baseada em padrões estabelecidos, desenhada de acordo com a necessidade de baixos custos de produção e sensores/actuadores sem fios de baixo consumo energético [34]. Esta tecnologia pode ser utilizada em muitos locais onde o controlo através de sistemas com fios seja impossível ou muito difícil (p. ex. locais de difícil acesso), é simples de operar e necessita de pouca energia, tornando-a actualmente uma inovação face aos actuais sistemas sem fios existentes no mercado.

Este sistema possui uma série de padrões e características [35]:

- Baseada na norma de comunicação IEEE 802.15.4;
- Utiliza bandas com frequências de 2.4 GHz (Mundial), 915 MHz (América) e 868 MHz (Europa);
- Providencia taxas de transferência de 250 Kbs @ 2.4 GHz (16 canais), 40 Kbs @ 915 MHz (10 canais) e 20 Kbs @ 868 MHz (1 canal);
- Distâncias de transmissão de 10 até 1600 metros, dependendo das condições do terreno (edifícios, montanhas, condições ambientais, etc.).

O sistema proposto é baseado numa estação base (Base Station) e alguns nós (Nodes) de monitorização [34].

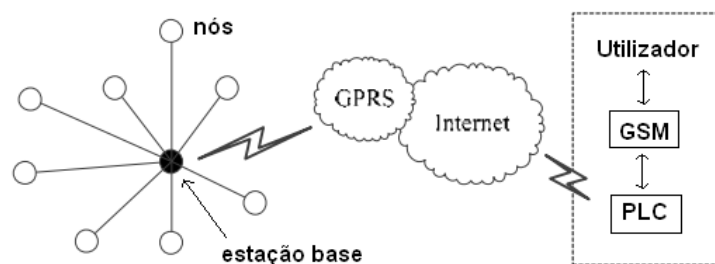


Figura 3.28 - Sistema proposto.

Estes nós são responsáveis pela recolha de dados relativos à temperatura, humidade e acidez do solo, enquanto a estação base é responsável pela coordenação entre os vários nós. A informação recolhida pelos nós é posteriormente enviada para a estação base via GPRS, onde será devidamente analisada e tratada pela CPU do PLC.

Por sua vez, a informação estará acessível ao utilizador via GSM (conexão GSM - PLC referida anteriormente) e será armazenada juntamente ou separadamente com os dados recolhidos pelo sistema implementado neste projecto.

A estação base é constituída por um dispositivo de monitorização *online*, uma unidade terminal de dados (DTU) GPRS e um módulo *ZigBee* (Fig. 3.29) [34].

O DTU GPRS implementa a transmissão de dados remota através da rede GPRS. O módulo *ZigBee* consiste num controlador MSP430 da *Texas Instruments* e um chip CC2420 RF. Este último é utilizado para recolher a informação dos nós e trocar informação com o PLC através do DTU via GPRS [34].

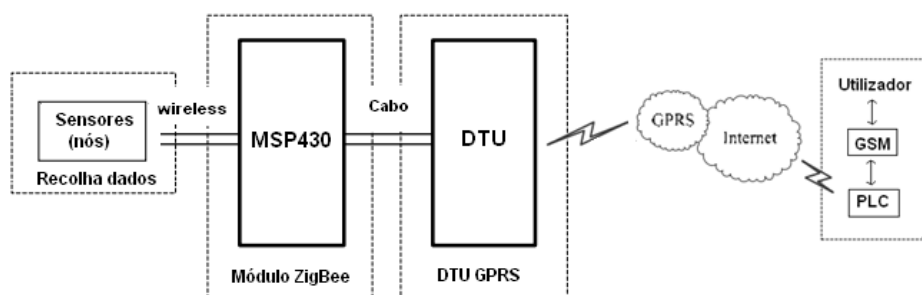


Figura 3.29 - Sistema proposto detalhado.

Ambos FFD (*Full Function Device* - dispositivo de funções completas que funciona como coordenador do módulo *ZigBee* da estação base) e RDF (*Reduced Function Device* - dispositivo de funções reduzidas referente ao módulo *ZigBee* presente nos nós de monitorização) são construídos com o sistema operativo *TinyOS*. Este OS é um sistema embutido para redes de comunicação sem fios que apresenta um conjunto de componentes necessários pré-incluídos para as aplicações [34].

O RDF possui duas funções: ler a temperatura, humidade e acidez do solo como sensor de recolha de informação para o PLC e enviar a informação para o FFD.

Por sua vez, o FFD possui as seguintes funções: receber a informação enviada pelo RDF, ler a informação do sensor de recolha do PLC e enviar as informações obtidas pelo RDF para a plataforma de gerenciamento remota (neste caso, o PLC).

Já no PLC, a informação recebida seria posteriormente analisada, seriam tomadas as devidas decisões e comandos por parte da CPU e seria posteriormente armazenada na base de dados interna do PLC (*data logs*).

Assim, com a combinação de sistemas de comunicação *ZigBee* e GPRS pode-se implementar um sistema de monitorização de informação em tempo real e posterior armazenamento de dados para fins estatísticos e comparativos.

Este sistema apresenta as vantagens de ser independente das infra-estruturas de rede actuais, flexível e bastante acessível economicamente face às restantes alternativas actuais.

3.1.6.3. Sincronização do relógio interno via GPS

Durante a realização deste projecto, foi encontrado um problema relativo ao relógio interno do sistema. Após algumas horas de funcionamento, o relógio interno do PLC atrasa entre 1 a 5 minutos. Contudo, e quanto maior fosse o tempo que o relógio interno do PLC estivesse a contar, o relógio atrasava ainda mais, podendo-se então admitir que o atraso será proporcional ao crescente tempo de operação do relógio interno do PLC. Para o sistema implementado neste projecto, e como o PLC estaria permanentemente ligado, o tempo de atraso ao fim de alguns meses seria bastante avultado, levando assim a erros na criação dos *data logs*, entre outros.

Para corrigir este problema existem duas soluções: efectuar uma sincronização do relógio interno do PLC via GPS ou através de servidores NTP.

Relativamente à solução referente ao GPS, o princípio de funcionamento seria o seguinte [36]:

- O PLC recebe uma trama²⁵ GPS do tipo RMC de acordo com a norma NMEA 0183²⁶;
- As tramas são lidas através do receptor GPS ligado à *interface* RS232 do módulo de comunicação CM1241 RS232 do PLC;
- O bloco de função “gps_rcv” recebe a informação de tempo enviada pelo GPS, converte-a automaticamente para o formato DTL (*Data and Time Long*) e actualiza o relógio interno do PLC quando for activada.

O bloco de função “gps_rcv” está desenhado para funcionar com receptores GPS padrão que estejam de acordo com a norma NMEA 0183.

As suas especificações são as seguintes [36]:

Tabela 3.6 - Tabela de especificações do bloco de função “gps_rcv” [36].

| Parâmetro | Configuração | Descrição |
|---------------------------|------------------|----------------------------|
| Taxa de transmissão | 38400 <i>bit</i> | - |
| Paridade | Não | - |
| <i>Bits</i> de informação | 8 <i>bits</i> | 8 <i>bits</i> por carácter |
| <i>Bits</i> de paragem | 1 <i>bit</i> | - |
| Controlo de fluxo | Não | - |

²⁵ *trama* - pacote de dados codificado de um determinado canal de comunicação.

²⁶ *NMEA 0183* - conjunto de especificações para dispositivos electrónicos de navegação.



Figura 3.30 - Sistema proposto para sincronização do relógio interno do PLC.

O equipamento necessário (além do PLC e do *software* de programação) para a implementação deste sistema é o seguinte [36]:

- Módulo de comunicação CM1241 RS232;
- Cabo *ethernet*;
- Cabo PG/PC da Siemens;
- Receptor GPS;
- Cabo conector para GPS (conversor de MD6 para RS232).

3.1.6.4. Sincronização do relógio interno via servidor NTP

Relativamente ao método por servidor NTP, não apresenta nenhum tipo de material adicional, apenas uma ligação do PLC à internet. Os servidores NTP permitem manter os relógios quer de um computador, quer de um dispositivo electrónico sempre com a hora certa e apresentam enorme exactidão.

Para o acerto do relógio interno do PLC com recurso a um servidor NTP é apenas necessário activar a função “*Time synchronization*” na configuração do PLC no *software* e introduzir os IP’s referentes aos servidores NTP que se deseja utilizar.

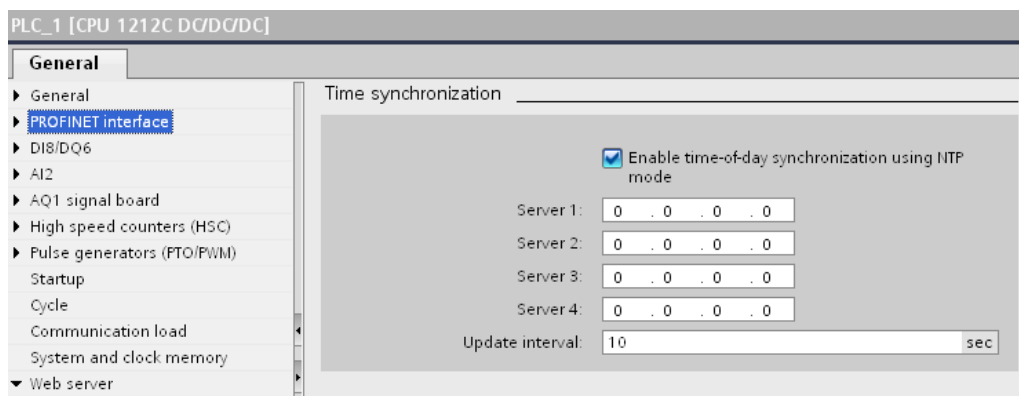


Figura 3.31 - Activação da sincronização do relógio interno via servidores NTP.

4. Conclusão

Com a realização deste projecto, pode-se afirmar que os principais objectivos do mesmo foram cumpridos, em particular a criação de um sistema de rega inteligente capaz de controlar e monitorizar um sistema de abastecimento de água, regular o pH da água do reservatório e sua posterior distribuição pelos diversos circuitos de rega do sistema. Foi também criada uma maquete com o intuito de simular todo o sistema, uma *interface* táctil, uma página *web* para um acesso, controlo e monitorização do sistema quer local quer remotamente e ainda implementado um sistema capaz de armazenar dados e valores referentes ao sistema (base de dados).

Este tipo de sistema apresenta grandes vantagens face a sistemas manuais para o mesmo objectivo, tais como, maior poupança energética, maior poupança de água, controlo preciso e eficaz das quantidades de água e valores do pH do tanque, possibilidade controlo local (consola HMI) e remoto (página *web*) e ainda a possibilidade de registo de valores para criação de uma base de dados. O sistema está também preparado para eventuais expansões e futuras melhorias, como por exemplo, a implementação de um sistema de controlo e monitorização *ZigBee* e GPRS e comunicação via GSM com o PLC.

Ao longo da implementação do projecto surgiram diversos problemas, nomeadamente a dificuldade de implementação e utilização do relógio interno do PLC e também problemas referentes à implementação do sistema de *data logging* (base de dados). Após várias pesquisas, chegou-se à conclusão que era necessário actualizar o PLC para o *firmware* 2.0 (no mínimo), para que fosse possível a implementação deste sistema.

Para efeitos de simulação e recorrendo à maquete elaborada, foram utilizados 6 botões de pressão ligados às entradas digitais do PLC que simulam o sensor de nível mínimo do reservatório, o sensor de nível máximo, o botão de paragem de emergência do sistema, o sensor de nível do reservatório do pH, o caudalímetro e o sensor de detecção de água na bomba. Relativamente às entradas analógicas, foram utilizados dois potenciómetros: um para simular o sensor de nível óptico (quantidade de água constante no tanque - 0 a 3000 litros) e outro para simular o valor do pH na água do tanque (0 a 14).

De forma a simular os avisos luminosos referentes aos alarmes do sistema, foram utilizados 3 LED's de cor vermelha e um LED de cor verde ligado à saída analógica do sistema, onde se pode observar a variação da intensidade luminosa do LED consoante a alteração do valor do pH. Esta variação da intensidade luminosa deve-se ao facto da saída analógica variar de 0 a 10V, e portanto, o LED estará apagado quando a alimentação for de 0V e estará aceso e com a luminosidade mais intensa quando a alimentação for de 10V.

Como o PLC não dispunha de saídas suficientes, o alarme luminoso referente à manutenção da bomba de água (<48 horas restantes para manutenção) foi implementado com recurso a um *bit* de memória e não a uma saída física. O mesmo aconteceu com o agitador mecânico, que, por falta de saídas disponíveis, foi implementado com recurso a um *bit* de memória.

Com a realização deste projecto posso concluir que enriqueci em muito o meu conhecimento referente aos PLC's e mais especificamente, em relação à programação do PLC S7-1200 da Siemens e consola HMI KTP600 Mono PN, adquirindo assim as bases necessárias para continuar a trabalhar e desenvolver projectos nesta área. Ganhei também bastante conhecimento e experiência relativamente à criação e programação de páginas *web* com recurso à linguagem HTML e programa *Notepad++*.

4.1. Trabalhos futuros

Após a conclusão desta dissertação é possível indicar vários trabalhos/melhorias futuras que podem ser desenvolvidas para melhorar o sistema elaborado no âmbito desta dissertação:

- (1) Implementação de um sistema de comunicação GSM/GPRS.
- (2) Implementação de um sistema de monitorização do solo via *ZigBee* e GPRS.
- (3) Implementação de um sistema de sincronização do relógio interno via GPS ou servidor NTP.

Estes trabalhos futuros podem ser considerados como uma mais-valia a implementar, para que este sistema preencha efectivamente todas as lacunas de um sistema manual do género e se melhore/simplifique a forma como o utilizador interage com o sistema.

5.Referências

- [1] L.A. Bryan and E.A. Bryan, *Programmable Controllers, Theory and Implementation - Second Edition*, An Industrial Text Company Publication, USA, 1997.
- [2] “Assobrav” [Online]. Disponível em:
<http://www.assobrav.com.br/imagens/noticia/noticias2243.jpg>.
- [3] “Funverde” [Online]. Disponível em:
http://farm5.static.flickr.com/4023/4367138153_8b85d5b1c2.jpg.
- [4] Marco António Ribeiro, *Automação Industrial - 4ª edição*, Salvador, BA, Outono de 2001.
- [5] “Blog” [Online]. Disponível em:
http://2.bp.blogspot.com/_jUDQadWqvo/R5UI7hMgxbI/AAAAAAAAABVE/Lx7jLJq-pTs/s400/praiado-calhau.jpg.
- [6] “M & S” [Online]. Disponível em:
<http://www.manutencaoesuprimentos.com.br/imagens/funcoes-de-um-separador-de-areia.jpg>.
- [7] “Perito.Med” [Online]. Disponível em:
http://4.bp.blogspot.com/_xpCb6C1Hnp4/TK0fk0f10FI/AAAAAAAAAAk/kyaDRLEK3lg/s320/HONDA.jpg.
- [8] Toni dos Santos Alvez, *Automação Industrial I*, Escola Superior de Tecnologia de Abrantes e Instituto Politécnico de Tomar, Departamento de Engenharia e Gestão Industrial - DEGI, 2004/2005.
- [9] Fernanda Esteve Gomes, *Soluções em Automação para Eficiência Energética*, Universidade Federal de Goiás, Escola de Engenharia Elétrica, 2003.
- [10] Prof. Sílvio José Pinto Simões Mariano e Prof. Pedro Miguel Figueiredo Oliveira Gaspar, Universidade da Beira Interior.
- [11] Wander Samuel Maass, *Automação de um forno para tratamento de chapas com controle via CPL e sistema supervisório*, Universidade Regional de Blumenau, Centro de Ciências Exatas e Naturais - Curso de Ciências da Computação, Dezembro de 2000.

- [12] Schuster, M., *Plug-in type connection techniques on encapsulated components on high-voltage equipment up to $U_m = 245$ kV*, Pfisterer Kontaktsysteme GmbH & Co. KG, 2005.
- [13] “Calibracon” [Online]. Disponível em:
<http://www.calibracon.com.br/produtos/umidade01.jpg>.
- [14] “Josmaq” [Online]. Disponível em:
http://www.josmaq.com.br/i/peças/sensor_de_temperatura+.jpg.
- [15] Robert H. Walden. *Analog-to-digital converter survey and analysis*, IEEE Journal on Selected Areas in Communication, April 1999.
- [16] “Siemens - Micro Automação” [Online]. Disponível em:
https://www.swe.siemens.com/portugal/web_nwa/pt/PortalInternet/QuemSomos/negocios/Industry/IA_DT/AutomationSystems/PublishingImages/S7%201200jpg.jpg.
- [17] “Controle e Monitoramento Inteligente” [Online], Disponível em:
<http://clpredes.files.wordpress.com/2010/06/clp-fig-4.jpg?w=480&h=200>.
- [18] “Electrónica - Relé” [Online]. Disponível em:
<http://www.electronica-pt.com/imagens/rele/rele.gif>.
- [19] “Electrónica - Transistor” [Online]. Disponível em:
<http://www.electronica-pt.com/index.php/content/view/47/37/>.
- [20] “Electrónica - Triacs” [Online]. Disponível em:
<http://www.electronica-pt.com/index.php/content/view/131/37/>.
- [21] “Direct Industry” [Online]. Disponível em:
http://img.directindustry.es/images_di/photo-m2/micro-automata-programable-528891.jpg.
- [22] “Desvendando os bits e bytes” [Online]. Disponível em:
<http://1.bp.blogspot.com/-Bk60Dxu6gkQ/TiHQ4BI2JZI/AAAAAAAAAD4/yY8HxGSPKeE/s1600/rom.jpg>.
- [23] “SBGadget” [Online]. Disponível em:
<http://www.sbgadget.com/wp-content/uploads/2007/11/step7.jpg>.

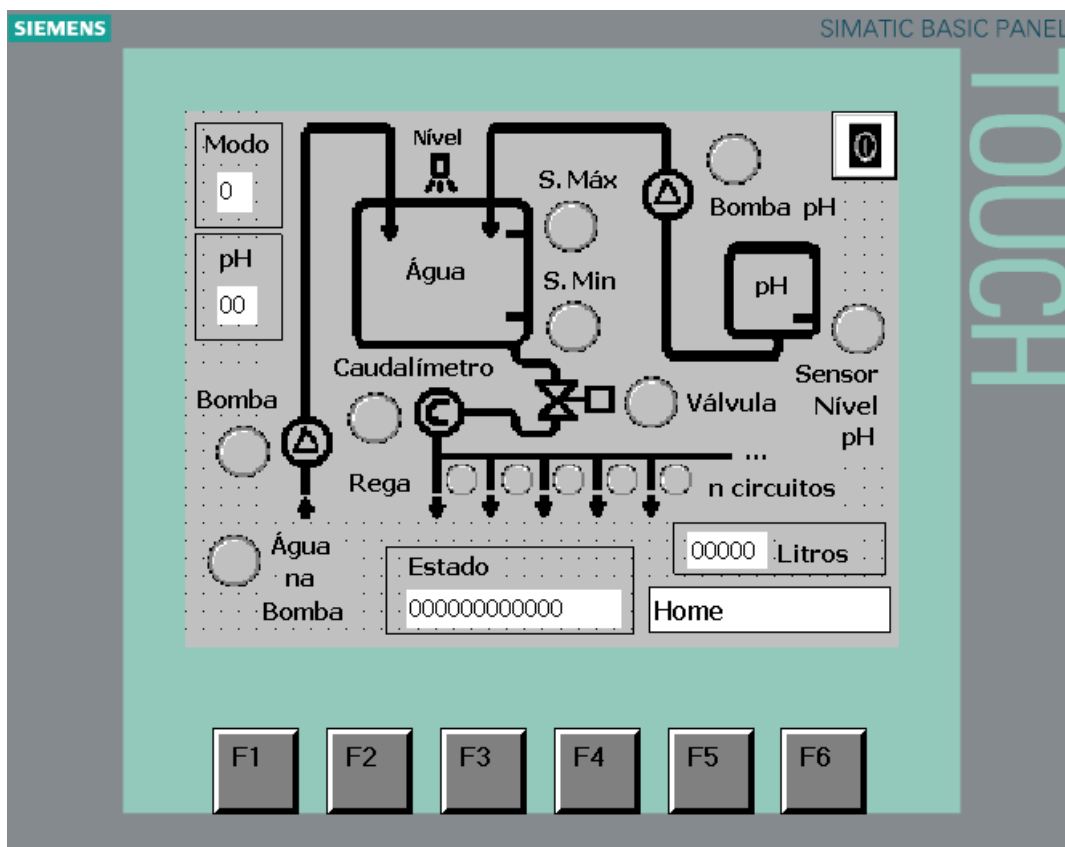
- [24] “Diotronic” [Online]. Disponível em:
<http://www.diotronic.com/imgs/dip-20.jpg>.
- [25] “Rockwell Automation - What is IEC 1131)” [Online]. Disponível em:
<http://www.rockwellsoftware.com/corporate/reference/iec1131/>.
- [26] William S. Vianna, *Controlador Lógico Programável*, CEFET - A educação tecnológica do ano 2000.
- [27] Siemens Simatic, *S7-1200 Programmable Controller System Manual*, Novembro de 2011.
- [28] Prof. Sandro Rodrigo G. Bastos, *Sistemas Digitais I*, Universidade Santa Cecília.
- [29] Siemens Simatic HMI, *HMI Devices Basic Panels*, Abril de 2012.
- [30] Esteban Egea-Lopez, Alejandro Martinez-Sala, Javier Vales-Alonso, Joan Garcia-Haro, Josemaria Malgosa-Sanahuja, *Wireless communications deployment in industry: a review of issues, options and technologies*, Computers in Industry 56 (2005) 29-53, Spain, 13 December of 2004.
- [31] S. Redl, M. K. Weber, M. Oliphant, *An Introduction to GSM*, The Artech House of Mobile Communications, 1995.
- [32] A. Alheraish, W. Alomar and M. Abu-Al-Ela, *Remote PLC system using GSM network with application to home security system*, 18th National Computer Conference, Saudi Arabia, 2006.
- [33] Siemens SIMATIC NET, *S7-1200 - Telecontrol CP 1242-7*, Novembro de 2011.
- [34] Xin Wang, Longquan Ma, Huizhong Yang, *Online Water Monitoring System Based on ZigBee and GPRS*, Procedia Engineering 15 (2011) 2680-2684, China, 2011.
- [35] “ZigBee Alliance - Standards” [Online]. Disponível em:
<http://www.zigbee.org/About/AboutTechnology/Standards.aspx>.
- [36] “Siemens Industry Online Support” [Online]. Disponível em:
<https://support.automation.siemens.com/WW/llisapi.dll?func=cslib.csinfo2&aktprim=99&lang=en>.

6. Anexos

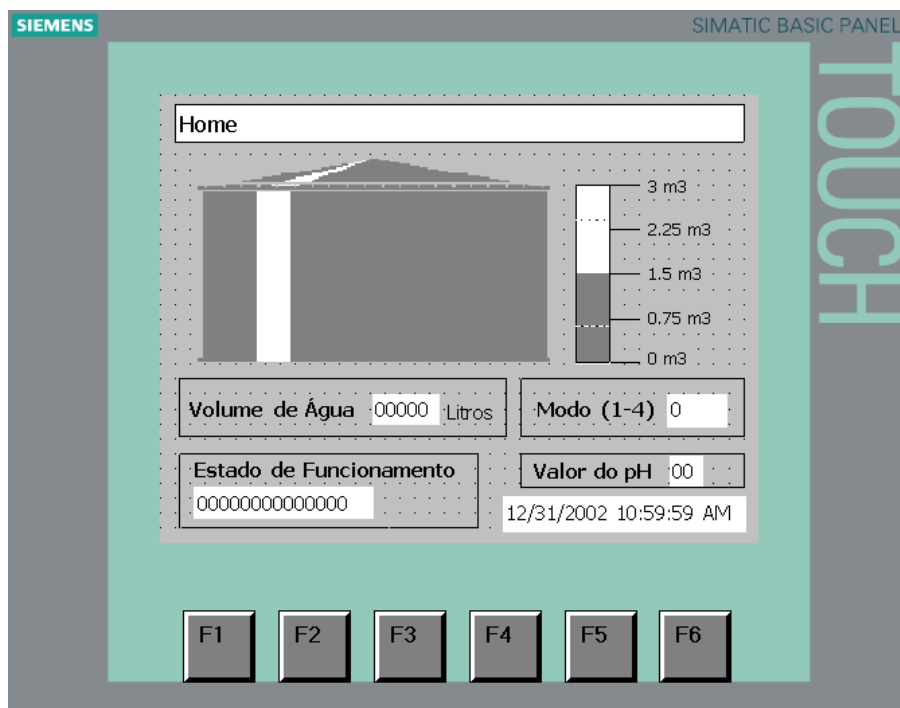
6.1. Anexo I

Neste anexo são apresentados e explicados todos os menus referentes ao programa da consola HMI e que permitem o total controlo e monitorização do sistema.

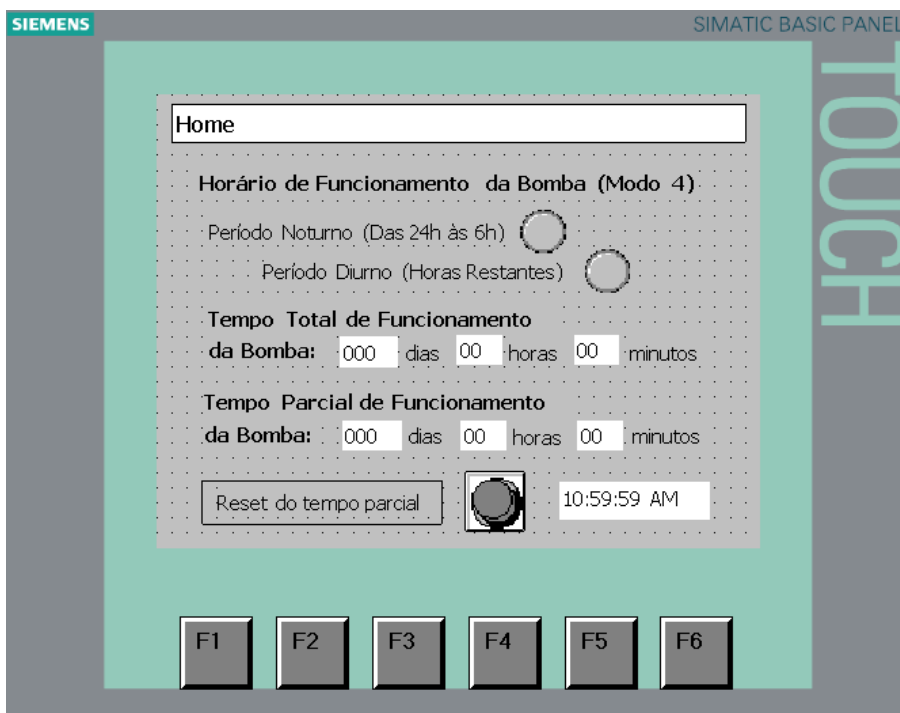
NOTA: O utilizador só pode fazer qualquer alteração no sistema caso esteja identificado como administrador (*login* de administrador). Caso contrário, pode visualizar mas não alterar. Em todos os menus está presente uma barra de navegação (limite superior do menu, excepto no menu “Home” que se encontra no canto inferior direito) que permite trocar e navegar através de todos os menus da consola.



No menu “Home” (menu inicial pré-definido) o utilizador pode consultar o estado de funcionamento de todo o sistema, incluindo o estado dos componentes, modo de funcionamento, estado de funcionamento, quantidade de água no tanque e ainda o valor actual do pH.



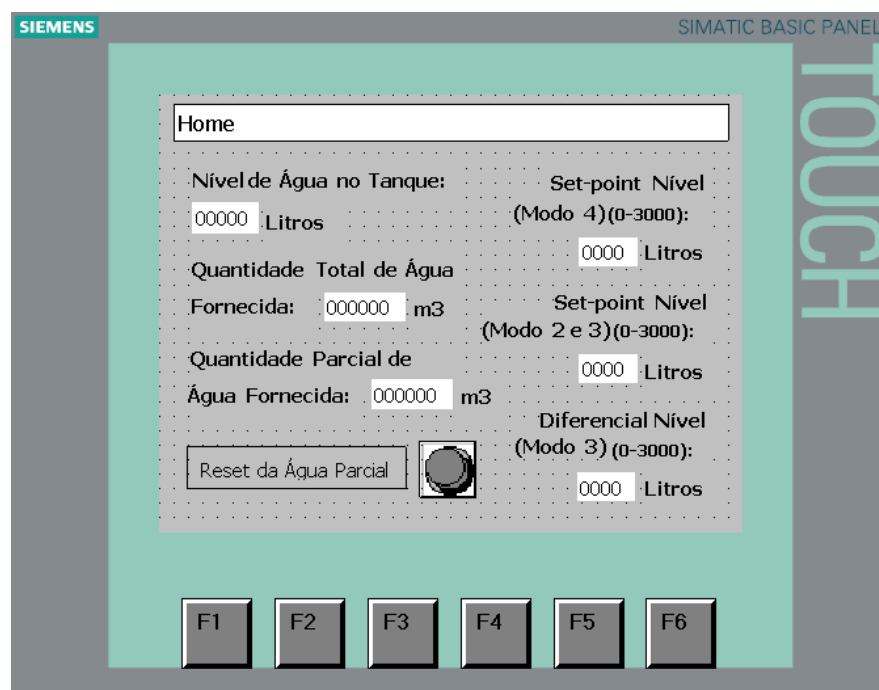
No menu “Visualização Geral” o utilizador tem uma visão geral de todo o sistema. Pode visualizar a quantidade de água no tanque a cada instante, o modo de funcionamento actual, o estado de funcionamento do sistema, o valor do pH e ainda a data e relógio interno do sistema.



No menu “Configurações de Horário” o utilizador tem a possibilidade de consultar, quando no Modo 4 de funcionamento, se o sistema se encontra a funcionar no Modo Nocturno ou Diurno, consultar o tempo total e parcial de funcionamento da bomba de água e ainda fazer *reset* ao tempo parcial.



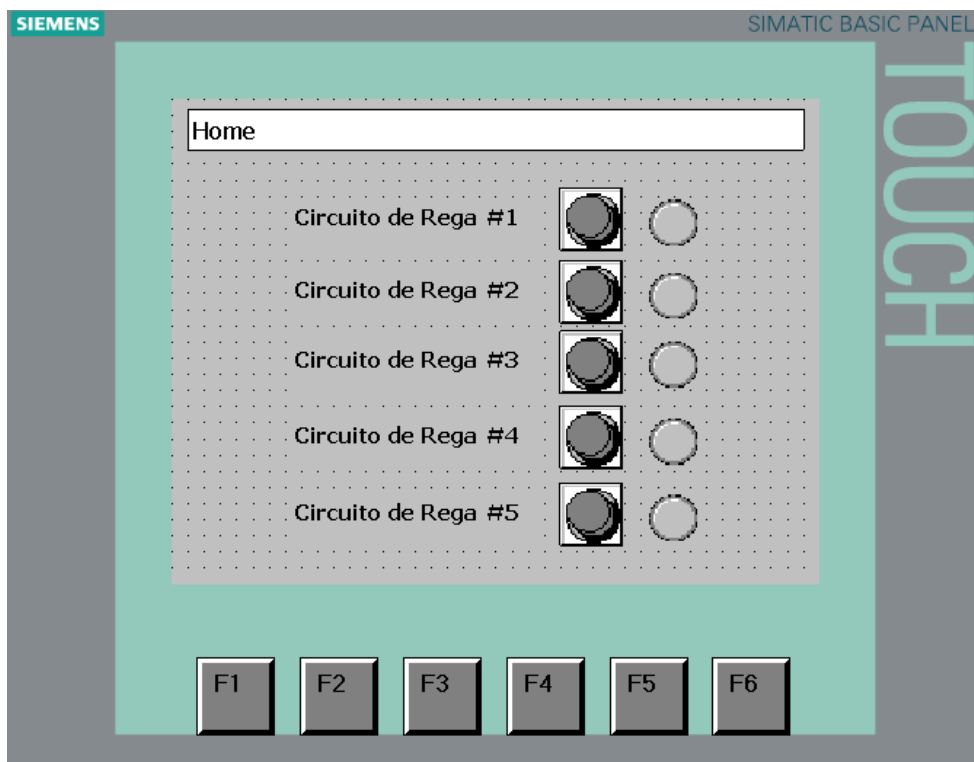
No menu “Configurações do PH”, e como o próprio nome indica, o utilizador pode consultar os valores do set-point do pH, o valor de alerta e alarme do pH, a velocidade da bomba do pH (apenas tem efeito no Modo 1) e ainda activar ou desactivar o controlo automático do pH.



No menu “Quantidades e Níveis” o utilizador pode consultar e alterar o valor do set-point do nível de água do Modo 2 e 4 e ainda o valor do diferencial do Modo 3. Pode também consultar a quantidade de água actual do tanque, bem como as quantidades total e parcial de água fornecidas pelo sistema. É possível neste menu fazer o *reset* da quantidade de água parcial fornecida.



No menu “Modo Manual” o utilizador tem a possibilidade de visualizar e controlar manualmente o estado de funcionamento da bomba de água, da válvula de saída e da bomba do pH. Pode ainda ligar ou desligar o controlo automático do pH.



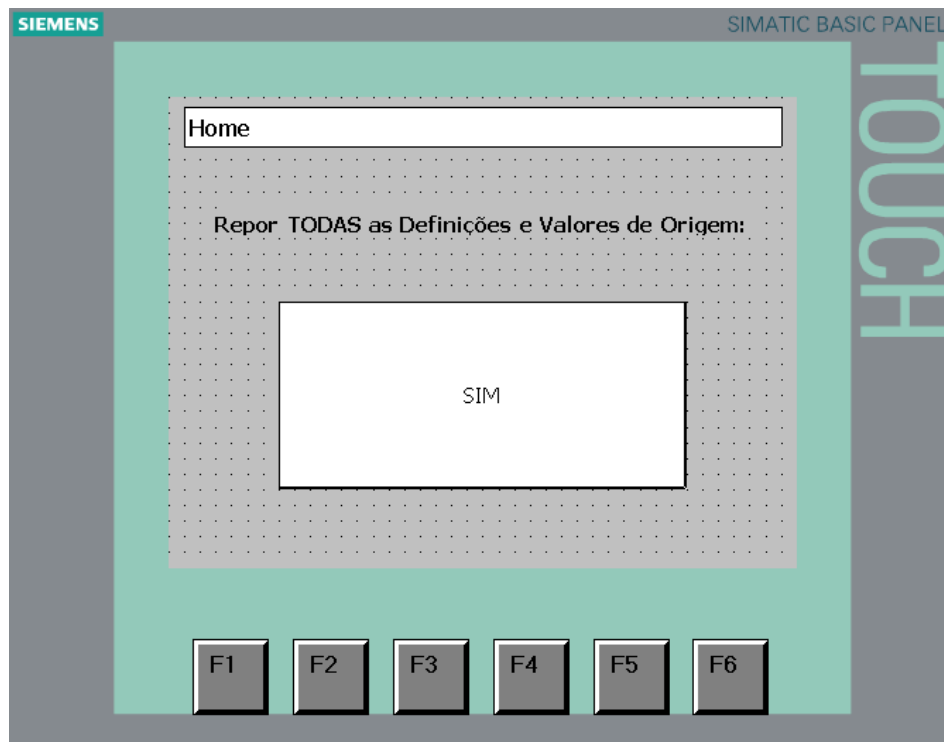
No menu “Controlo dos circuitos de Rega” o utilizador pode consultar e alterar o estado dos circuitos de rega do sistema a qualquer momento.



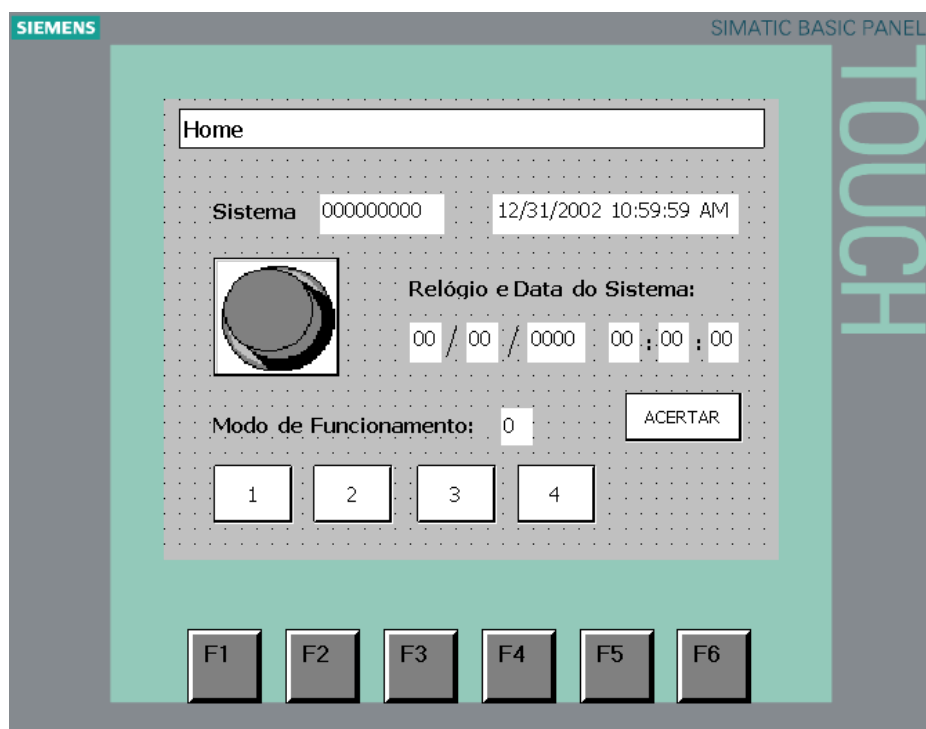
No menu “Manutenção” o utilizador pode consultar o tempo restante para a manutenção da bomba e ainda fazer *reset* ao mesmo.



No menu “Informação do Projecto” estão presentes algumas informações relativas à criação do projecto.



No menu “Definições de Origem” o utilizador tem a possibilidade de repor todas as definições e valores de origem do sistema.



No menu “Arranque e Funcionamento” o utilizador pode consultar e alterar o estado de funcionamento do sistema (ligado ou desligado), consultar e alterar a data e relógio interno do sistema e ainda consultar e alterar o modo de funcionamento do sistema.

6.2. Anexo II

Neste anexo é apresentada a página *web* criada para o controlo e monitorização do sistema apresentado neste projecto. Serão também apresentadas e explicadas todas as subpáginas da mesma.

The screenshot shows a web browser window with the address bar displaying '192.168.1.30 https://192.168.1.30/awp/Home/Visualizacao.htm'. The main heading of the page is 'Controlo e Monitorização de Sistema de Abastecimento de Água'. On the left, there is a sidebar menu with the following links: 'Visualização', 'Horário', 'Configurações PH', 'Quantidades', 'Controlo Manual', 'Circuitos de Rega', 'Manutenção', 'Definições de', and 'Origem'. The main content area is titled 'Visualização Geral do Sistema' and contains four sections of system data, each presented in a table-like format with a light blue header and white data rows.

| Informação do Sistema | |
|------------------------|----------------|
| Sistema: | Ligado |
| Estado do sistema: | Encher |
| Modo de Funcionamento: | 2 (Auto_Nivel) |
| Quantidade de Água: | 1888 Litros |
| Valor do PH: | 9 |

| Componentes do Sistema | |
|------------------------|-----------|
| Bomba de Água: | Ligada |
| Bomba do PH: | Desligada |
| Válvula: | Aberta |

| Circuitos de Rega | |
|-------------------|--------|
| Circuito #1: | Ligado |
| Circuito #2: | Ligado |
| Circuito #3: | Ligado |
| Circuito #4: | Ligado |
| Circuito #5: | Ligado |

| Alarmes | |
|----------------------------|---------------|
| Presença de Água na Bomba: | Sim |
| Nível do Depósito do PH: | Normal |
| Alarme do Valor do PH: | Desligado |
| Paragem de Emergência: | Desligado |
| Manutenção da Bomba: | Desnecessária |

Nesta página (página inicial), o utilizador tem acesso à visualização de todos os componentes, valores e alarmes do sistema.

192.168.1.30
https://192.168.1.30/awp/Home/Horario.htm

Controlo e Monotorização de Sistema de Abastecimento de Água

[Visualização](#)
[Horário](#)
[Configurações PH](#)
[Quantidades](#)
[Controlo Manual](#)
[Circuitos de Rega](#)
[Manutenção](#)
[Definições de](#)
[Origem](#)

Horário do Sistema

Poupança de Energia
Modo Nocturno (Das 24h até às 6h): Desactivado
Modo Diurno: Desactivado

Tempo de Funcionamento da Bomba
Tempo Total: 5 d 0 h 18 m
Tempo Parcial: 5 d 0 h 18 m

Na página “Horário” o utilizador tem acesso a consultar, quando o sistema está no Modo 4 de funcionamento (Nocturno), qual o modo onde o PLC se encontra: Modo Nocturno ou Diurno. Além disso, pode também consultar o tempo total e parcial de funcionamento da bomba. Por sua vez, tem a possibilidade de fazer *reset* ao tempo parcial.

192.168.1.30
https://192.168.1.30/awp/Home/Ph.htm

Controlo e Monotorização de Sistema de Abastecimento de Água

[Visualização](#)
[Horário](#)
[Configurações PH](#)
[Quantidades](#)
[Controlo Manual](#)
[Circuitos de Rega](#)
[Manutenção](#)
[Definições de](#)
[Origem](#)

Configurações do PH

Valores do Sistema
Set-Point do PH: 7
Valor de Alerta do PH: 4
Valor de Alarme do PH: 2
Controlo Automático do PH: Sim

Activado

Na página “Configurações PH” o utilizador pode consultar e alterar os valores do set-point do pH, valor de alerta do pH e valor de alarme do pH. Pode ainda activar ou desactivar o controlo automático do pH.

192.168.1.30 | https://192.168.1.30/awp/Home/Quantidades.htm

Controlo e Monitorização de Sistema de Abastecimento de Água

[Visualização](#)
[Horário](#)
[Configurações PH](#)
[Quantidades](#)
[Controlo Manual](#)
[Circuitos de Rega](#)
[Manutenção](#)
[Definições de](#)
[Origem](#)

Quantidades

Quantidades

Quantidade Total de Água Fornecida: 40 m3

Quantidade Parcial de Água Fornecida: 40 m3

Níveis

Set-Point Nível de Água (Modo 2 e 3): Litros

Diferencial de Água (Modo 3): Litros

Set-Point Nível de Água (Modo 4): Litros

Na página “Quantidades” o utilizador pode consultar as quantidades total e parcial fornecidas e fazer *reset* à quantidade parcial de água. É-lhe também permitido consultar e alterar os valores do set-point referentes ao nível de água do Modo 2, valor do diferencial do Modo 3 e valor do set-point de nível do Modo 4. Caso seja introduzido um valor inferior a 0 e superior a 3000, surge uma mensagem de erro à direita do respectivo campo onde o erro foi introduzido.

192.168.1.30 https://192.168.1.30/awp/Home/Manual.htm

Controlo e Monotorização de Sistema de Abastecimento de Água

[Visualização](#)
[Horário](#)
[Configurações PH](#)
[Quantidades](#)
[Controlo Manual](#)
[Circuitos de Rega](#)
[Manutenção](#)
[Definições de](#)
[Origem](#)

Controlo Manual

Modo de Funcionamento Actual: 2 (Auto_Nivel)

Bomba de Água:

Bomba do PH:

Válvula de Água:

Velocidade da Bomba do PH: %

Estado do Sistema: Ligado

Na página “Controlo Manual” o utilizador pode consultar e alterar o modo de funcionamento actual do sistema, consultar e alterar (quando no Modo Manual) o estado da bomba de água, da bomba do pH, da válvula de água e a velocidade da bomba do pH. Pode também consultar e alterar o actual estado de funcionamento do sistema.

192.168.1.30 https://192.168.1.30/awp/Home/Circuitos.htm

Controlo e Monotorização de Sistema de Abastecimento de Água

[Visualização](#)
[Horário](#)
[Configurações PH](#)
[Quantidades](#)
[Controlo Manual](#)
[Circuitos de Rega](#)
[Manutenção](#)
[Definições de](#)
[Origem](#)

Circuitos de Rega

Modo de Funcionamento Actual: 2 (Auto_Nivel)

| | | |
|--------------|--------|--------|
| Circuito #1: | Ligado | Ligado |
| Circuito #2: | Ligado | Ligado |
| Circuito #3: | Ligado | Ligado |
| Circuito #4: | Ligado | Ligado |
| Circuito #5: | Ligado | Ligado |

Submeter

Na página “Circuitos de Rega” o utilizador pode consultar o modo de funcionamento actual do sistema e ainda consultar e alterar o estado dos circuitos de rega do sistema.

192.168.1.30 https://192.168.1.30/awp/Home/Manutencao.htm

Controlo e Monotorização de Sistema de Abastecimento de Água

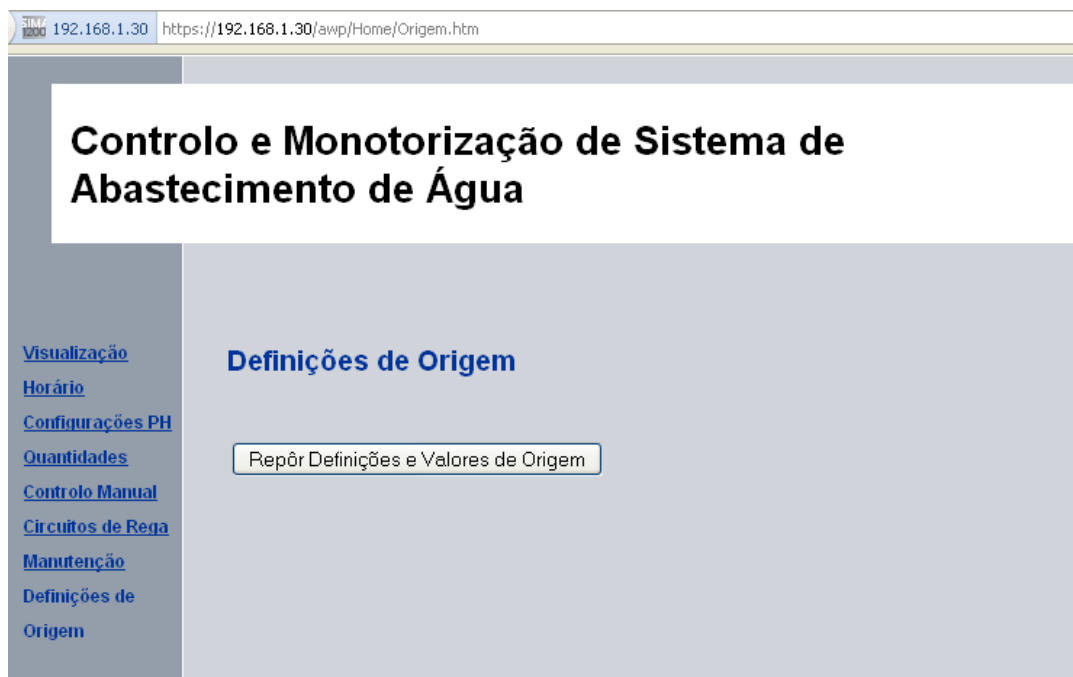
[Visualização](#)
[Horário](#)
[Configurações PH](#)
[Quantidades](#)
[Controlo Manual](#)
[Circuitos de Rega](#)
[Manutenção](#)
[Definições de](#)
[Origem](#)

Manutenção da Bomba

Tempo Restante para Manutenção da Bomba: 10000 horas

Reset Tempo de Manutenção

Na página “Manutenção” o utilizador pode consultar e fazer *reset* ao tempo de manutenção restante da bomba de água.



Por último, na página “Definições de Origem”, e como o próprio nome indica, o utilizador pode restaurar as definições e valores de origem do sistema.

NOTA: Assim como na consola, o utilizador só pode fazer qualquer alteração no sistema caso esteja identificado como administrador (*login* de administrador). Caso contrário, pode apenas visualizar mas não alterar.